Multi-Objective Optimization of the Punchiná reservoir in Colombia

Additional Thesis

Marisol Irías Mata

TU Delft

Multi-Objective Optimization of the Punchiná reservoir in Colombia

by

Marisol Irías Mata

Supervisors:

Dr. Ir. J. P. Aguilar López	*Delft University of Technology*
Dr. Ir. E. Abraham	*Delft University of Technology*
Msc. J. A. Villada Arroyave	*Delft University of Technology* *IHE Delft Institute for Water Education*

Additional Thesis
Master in Hydraulic Engineering

Delft University of Technology
The Netherlands
January 2021

Abstract

Punchiná reservoir is part of the San Carlos Hydroelectric Power Plant, situated in the Guatapé watershed. The Guatapé river is an affluent of the Samaná Norte river, which in turn is an affluent of the Magdalena river. San Carlos Hydroelectric Project uses the waters from the rivers San Carlos and Guatapé and discharges the turbined flow directly into the Samaná Norte river by a tunnel. Currently, there is flow downstream of the Punchiná dam only on the days where the spillway operates, significantly impacting the riverine ecosystem. Additionally, claims have been made about how hydropeaking causes floods in villages downstream, particularly in La Pesca village. This town is located in the confluence of Samaná Norte and Magdalena river, on the left bank of Samaná Norte river mouth.

The present report deals with the multi-objective optimization of the Punchiná reservoir of San Carlos Hydroelectric Project in Colombia by considering the objectives of maximizing hydropower revenues, maximizing the ecological discharge at Guatapé river, downstream of the dam, and reducing the flood risk at La Pesca village.

Four numerical methods were coded in Python to solve the reservoir routing. To solve the multi-objective optimization, the non-dominated sorting genetic algorithm II (NSGA-II) using the *Pymoo* framework in Python was set up, along with the use of an Explicit Euler numerical method for modelling the river routing. The simulation was performed for 3 periods (high, average and low flow conditions) within the years 2010-2017.

After multiple optimization scenarios, it can be concluded that the hydropower and environmental flows are competing objectives, i.e., allocating water for environmental flow purposes from the Punchiná reservoir will always result in a reduction of the hydropower revenues. Hence, it is recommended that an incentive system is developed so that the ecosystemic services are compensated to persuade the generating companies into including ecological objectives into their optimal operation curves. In addition, suggestions on considering a bypass tunnel to let the discharge flow into Guatapé river dry trajectory while adding a turbine to take advantage of this flow are also given.

The results also show that the flood mitigation objective does not result in a competing objective against the hydropower and environmental flow objectives when there are average flow conditions in the Magdalena river. Floods commonly occur during extreme weather periods whereas the optimization of the Punchiná reservoir is performed for monthly average flow conditions at Magdalena river. Thus, to assess the hydropeaking effect in the water levels at La Pesca site, it is recommended that the reservoir optimization should also include extreme flow conditions at Magdalena river when experienced.

Contents

1 Introduction

Reservoirs are water impoundments intended for multiple purposes. They can be used for hydropower, irrigation, flood mitigation or recreation. In some cases, a reservoir fulfils only one of the abovementioned purposes while many other times is created to satisfy different human necessities simultaneously. Although reservoirs provide many benefits for society, the negative effects on the natural flow regime downstream and upstream, and the impacts on biodiversity are well-established. Moreover, the inputs of a reservoir are often uncertain and therefore conceived as stochastic. To achieve an optimal operation of the reservoirs many factors come into play. An optimal outcome depends on the meteorological conditions and on which reservoir functions are more important for the region and the reservoir managers. Is it better to generate the most amount of revenues neglecting the effect of high discharges downstream? Should the water be stored during the wet months in case there is not enough inflow during the dry season? Should the flora and fauna of the reach downstream of the dam be considered while operating a reservoir? These and many other questions are faced by the reservoir operators on a daily basis. And the answers are not so easy to obtain.

Multi-objective optimization provides a decision-making framework to help answering the above questions. The present report aims to solve the multi-objective optimization of the Punchiná reservoir of San Carlos Hydroelectric Project in Colombia and is part of the PhD dissertation project of Msc. Jairo Villada intitled *Implementation of environmental flow regime for multicomponent hydropower generation*. Colombia has the 6th cleanest electricity generation mix in the world with 68% of the installed capacity generated by renewable resources, mainly from hydropower (*Acolgen - Colombian Association of Electric Power Generators,* 2019).

1.1 Problem description

Punchiná reservoir is part of the San Carlos Hydroelectric Power Plant, situated in the Guatapé watershed. The Guatapé river is an affluent of the Samaná Norte river, which in turn is an affluent of the Magdalena river. San Carlos Hydroelectric Project uses the waters from the rivers San Carlos and Guatapé and discharges the turbined flow directly into the Samaná Norte river by a tunnel (see Figure 1-1).

Currently, the Punchiná reservoir, located along the Guatapé river, is operated for hydropower generation as its main objective and does not contain a structure that allows continuous flow (environmental flow) downstream of the dam. For over 30 years, the Guatapé river has experienced a disruption of its natural flow caused by San Carlos Hydroelectric Project. Nowadays, there is only flow downstream of the dam on the days when the spillway operates, impacting largely the riverine ecosystem. A few consequences on the river ecosystem are that the flora and fauna both downstream and upstream had experienced a permanent alteration due to a complete change of the natural environment. Since the dam acts as a barrier for migratory river animals, this structure impedes fishes to swim upstream and even downstream, endangering the species. Moreover, permanent inundation upstream reduces the environment dynamics, reducing wildlife diversity. The temperature of the water is also modified on the reservoir area, affecting the aquicolous species. Besides, by changing the flood character of the river, the marsh landscape is reduced and by trapping sediment and debris, the loss of nutrients and habitat for animals downstream is also diminished (Lin, 2011). Many more impacts are experienced by the riverine ecosystem as consequence of river damming.

In addition, concerns have been cast about how hydropeaking is one of the causes of floods in villages along the Samaná Norte river, more specifically, in La Pesca village. This town is located in the confluence of Samaná Norte and Magdalena river, on the left bank of Samaná Norte river mouth (see Figure 1-1).

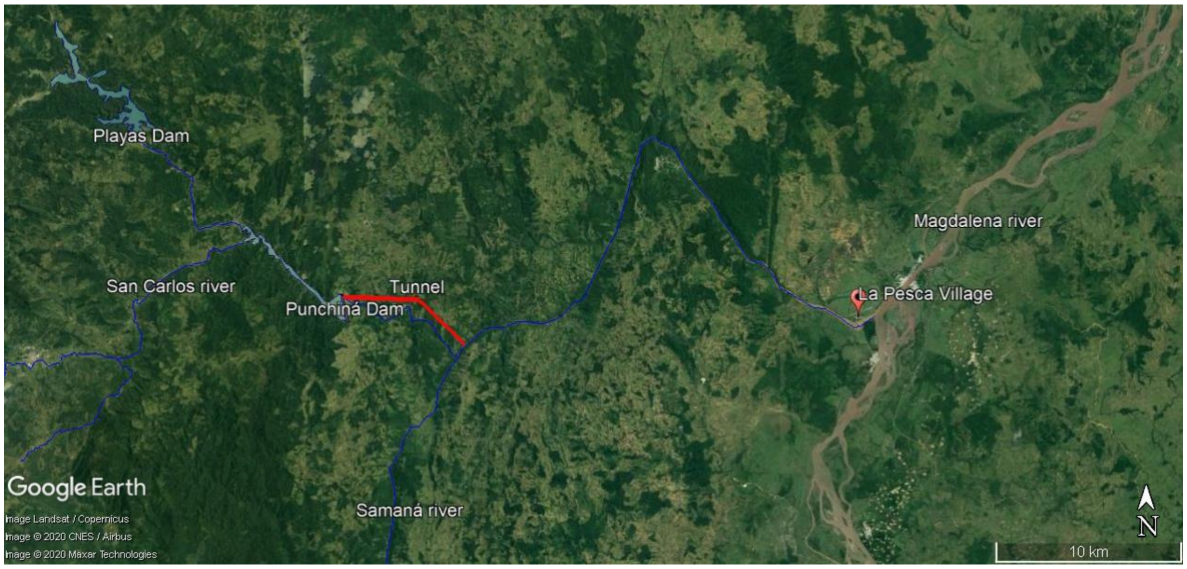


Figure 1-1. Punchiná reservoir System
Source: Google Earth

1.2 Objective and research questions

The aim of this study is to optimize the operation of the Punchiná reservoir by adding two more objectives to its operation goals: providing ecological discharge in Guatapé river, downstream of the dam, and mitigating the flood risk at La Pesca village, downstream of the powerhouse.

Hence, the main objective is formulated as:

To determine optimal operational rules for the Punchiná reservoir while considering hydropower production, ecological discharge and flood mitigation

Multi-objective optimization problems usually involve the solution of conflicting objectives, implying there is not a single optimal solution that satisfies all the conditions imposed. The Pareto-optimal solutions are the outcome of the optimization and allow the exploration of the trade-offs between the different objective functions.

Based on the outlined problem and the aim of the project, this study focuses on answering the following research question and sub-questions:

What are the trade-offs between hydropower revenues, flood mitigation and ecological discharge in Punchiná reservoir?

What is the allowable ecological discharge that can be taken from Punchiná reservoir without impacting the hydropower generation?

How can the ecological discharge be provided in the Guatapé river, reach downstream of Punchiná dam?

How should the San Carlos Power Plant be operated to mitigate the flood risk at La Pesca village?

1.3 Project outline:

The structure of this report is as follows:

Chapter 1 states the problem description, main objective and research question of this study.

Chapter 2 gives an overview of the case study: Punchiná reservoir and San Carlos Hydroelectric Power Plant. It includes its location, the available data, the metereological conditions during the past decade and the situation at La Pesca village – situated at the Samaná Norte river mouth.

Chapter 3 explains the reservoir routing, along with four numerical methods coded in Python. The code developed in Python is explained in detail in Appendix 3: Description of Python Script to be further used by students interested in hydropower projects. This chapter also includes an outline of the multi-objective optimization algorithms, followed by a description of the optimization method applied to the case study. Finally, it deepens in the formulation of the multi-objective problem for the specific case by defining the objective functions and constraints of the system.

Chapter 4 includes the results and discussion and Chapter 5 gives recommendations to be implemented further along in the overall research of the Samaná Norte river. It also provides conclusions for the present research project.

2 Case study: Punchiná reservoir and San Carlos Hydroelectric Power Plant

2.1 General description

Punchiná reservoir is part of the San Carlos Hydroelectric Power Plant, situated in the Guatapé watershed. This project is "located near the district of El Jordán in the municipality of San Carlos, department of Antioquia, 150 kilometers east of Medellín" (ISAGEN, 2020). See Figure 2-1.

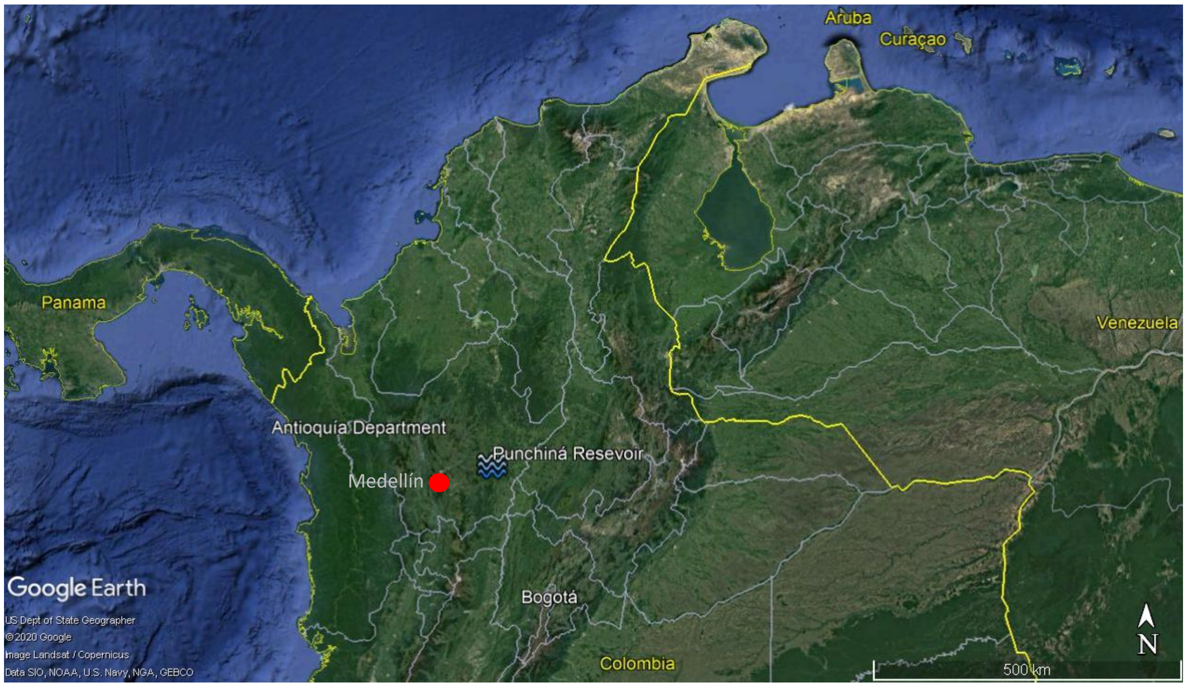


Figure 2-1. Location of Punchiná reservoir
Source: Google Earth

San Carlos Project uses the waters from the rivers San Carlos and Guatapé and discharges the turbined flow into the Samaná Norte river by a tunnel as shown in Figure 2-2. The spill discharge flows into the Guatapé river, in the reach located between the dam and the Samaná Norte river.

Upstream of the Guatapé river is Playas Hydroelectric Power Plant. Playas reservoir receives the outflows from two hydroelectric projects upstream, Guatapé and Jaguas Power Plants. Both projects intervene the Nare river, which is situated in the Nare Watershed. Thus, the natural system of Nare and Guatapé basins was modified by the hydroelectric development of this area. Due to the current system, flow is taken from the Nare watershed and is given to the Guatapé watershed. The Samaná Norte basins system is illustrated in Figure 2-3.

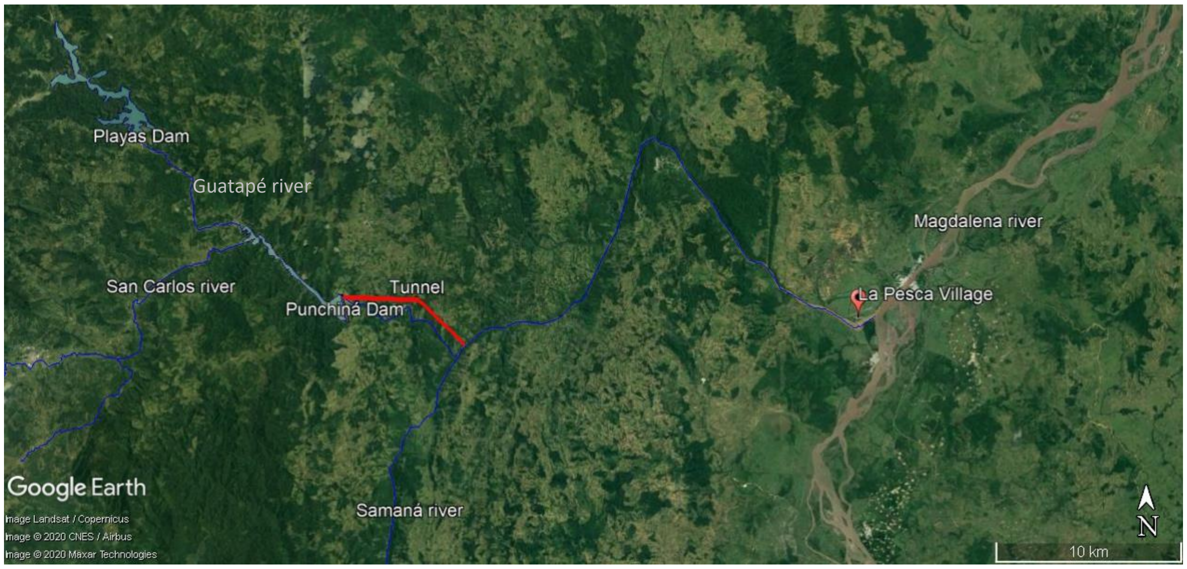


Figure 2-2. Punchiná reservoir System
Source: Google Earth

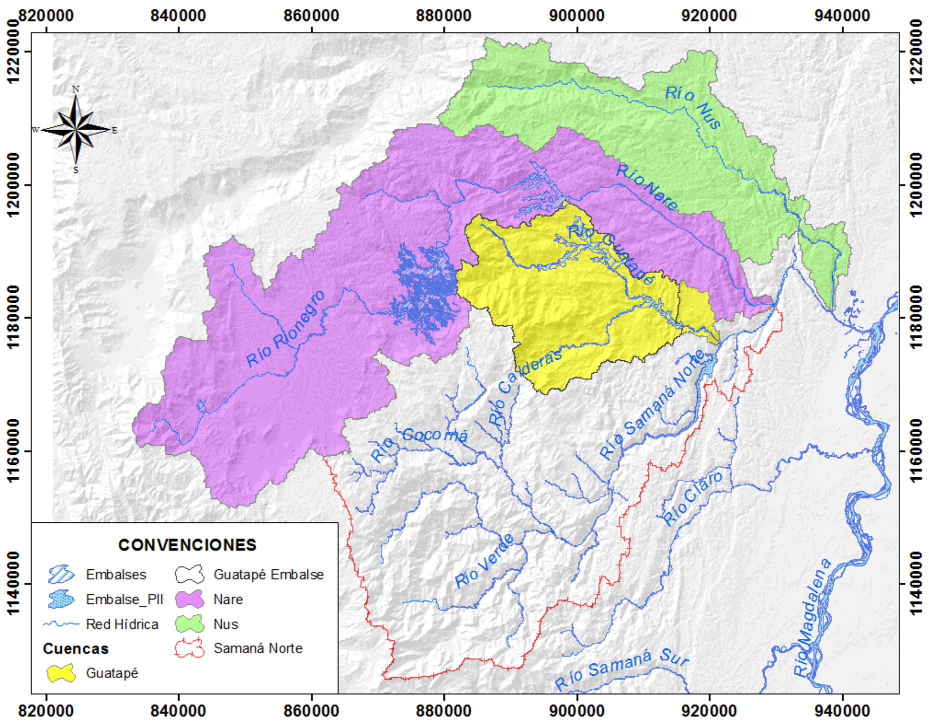


Figure 2-3. Watersheds of Samaná Norte River
Source: Integral - Proe S.A.S., 2013

The main characteristics of the San Carlos Hydroelectric Power Plant are summarized in Table 2-1.

Variable	Value	Units
Installed Capacity	1240	[MW]
Number of turbines	8	[-]
Installed Capacity per turbine	155	[MW]
Dam type	Earthfill	[-]
Dam crest level	781	[masL]
Minimum operation level	754	[masL]
Type spillway	Chute Spillway	[-]
Spillway crest level	775	[masL]
Maximum spillway capacity	7200	[m^3/s]
Chute width	Beginning: 130 End: 60	[m]

Table 2-1. Characteristics of San Carlos Hydroelectric Project
Source: ISAGEN

Two bathymetric studies were carried out in the years 2015 and 2019 by Batimetría S.A.S. The results reveal a change in the reservoir capacity and in the minimum water level. The latter decreased from 759 masL (meters above mean sea level) to 754 masL. The query was done, indicating that this change was caused by the sedimentation of the reservoir; nonetheless, the minimum design water level is 754 masL. The cross section of the earthfill dam is shown in Figure 2-4.

Figure 2-5 shows the reservoir capacity curves for both years. In the year 2019 the capacity for the water levels higher than 765 masL is a little bit larger than in the year 2015. It is considered that landslides occurred during the gap years, increasing the reservoir capacity.

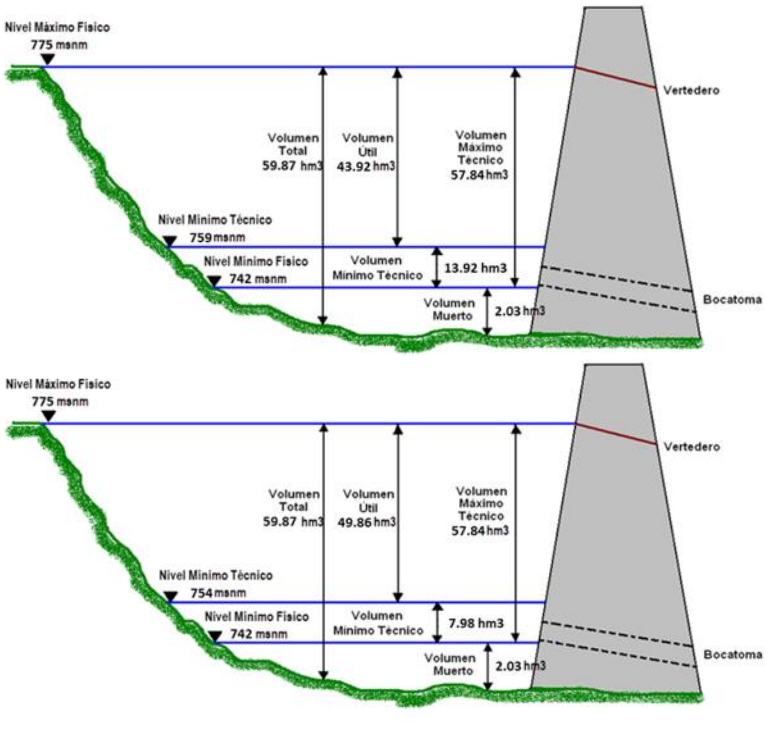
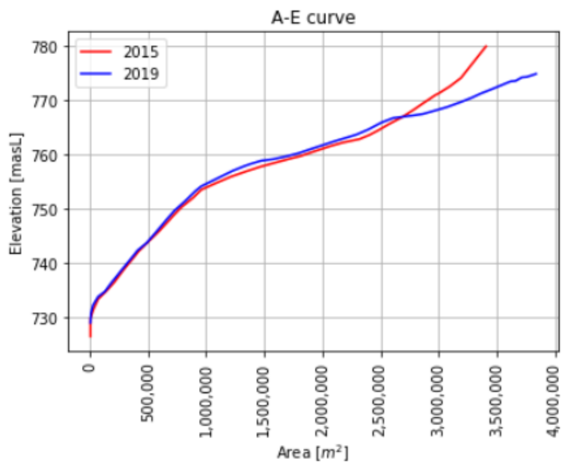


Figure 2-4. Cross-section of the earthfill dam. Top image: 2015 bathymetry. Bottom image: 2019 bathymetry.
Source: Batimetría S.A.S., 2019



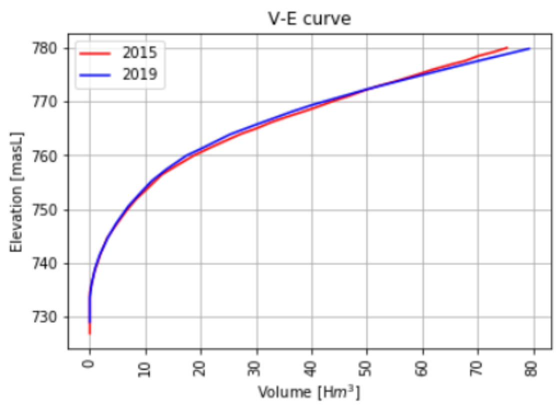


Figure 2-5. Punchiná reservoir capacity curves. Left image: Area. Right image: Volume.
Source: Batimetría S.A.S., 2019

2.2 Available Data

The data provided for the study is summarized in Table 2-2 and it includes time series for the period 2000-2017, conversion factor equations (Equations (1) and (2)) and reservoir capacity curves for Punchiná and Las Playas (Figure 2-5 and Figure 2-6 respectively).

Type of Data	**Period**	**Variables**	**Source**
Time series	2000-2017	• San Carlos discharge in m^3/s • Generated discharge in MWh/day for Punchiná and Playas resevoirs • Spill discharge in MWh/day and in m^3/day for Punchiná and Playas resevoirs • Reservoir Volume in m^3/day in Punchiná and Playas reservoir • Energy prices in \$/kWh (\$ refers to Colombian peso)	XM, (s. f.)
	2012-2017	• Water levels in masL at Punchiná reservoir	XM, (s. f.)
Equation	[-]	• Conversion factors equation for Punchiná reservoir • Conversion factors equation for Las Playas reservoir	Consejo Nacional de Operación, (2018) Consejo Nacional de Operación, (2019)
Paired Data	[-]	• Punchiná reservoir capacity curve • Las Playas reservoir capacity curve	Batimetría S.A.S., (2019) Díaz Serna, (2011)

Table 2-2. Available Data

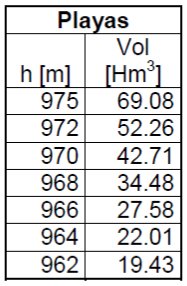
Conversion factors equation for both reservoirs (in Appendix 1: Case Study Data the conversion factor curves are displayed).

- Punchiná reservoir

$$FC\ (Elev) = -0.0001984389 * Elev^2 + 0.3155122110 * Elev - 119.9051413867\ [MW/(m^3/s)\,] \quad (1)$$

- Playas reservoir

$$FC\ (Elev) = -7.23826701490 + 0.0092138888 * Elev\ \ [MW/(m^3/s)\,] \quad (2)$$



Playas	
h [m]	Vol [Hm^3]
975	69.08
972	52.26
970	42.71
968	34.48
966	27.58
964	22.01
962	19.43

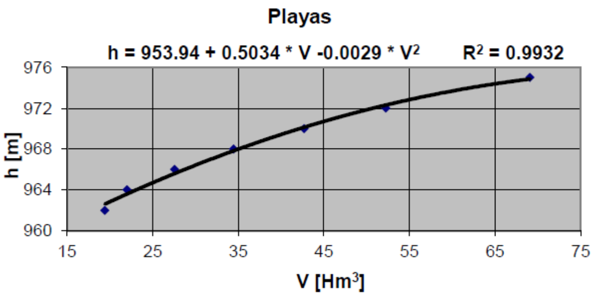


Figure 2-6. Playas reservoir curve
Source: Díaz Serna, 2011

2.3 Guatapé Discharge – Downstream of Punchiná Dam

The Guatapé Watershed downstream of Punchina dam has an area of 38.40 km^2 and the river reach has a length of 48 km (Integral - Proe S.A.S., 2013). Due to the human interventions in the Guatapé watershed as a result of the hydroelectric projects, the Guatapé river downstream of the Punchina reservoir is regulated almost completely by the operation of the projects upstream. Currently, the flow is quite constant along the year ranging from 2.0 m^3/s until 17.8 m^3/s, with higher discharges in the rainy months of September, October and November due to higher flows over the spillway of the San Carlos Project (Table 2-3 and Table 2-4).

Month	Jan	Feb	Mar	Apr	May	June	July	Aug	Sep	Oct	Novr	Dic
Guatapé River	3.36	3.07	2.00	7.88	6.01	7.61	6.53	9.65	10.50	17.84	15.03	10.00

Table 2-3. Mean discharges in Guatapé river, reach downstream of Punchiná reservoir
Source: Integral - Proe S.A.S., 2013

Tr (Años)	2,33	5	10	25	50	100
Minimum discharges (m^3/s)	0,823	0,683	0,594	0,505	0,453	0,410
Maximum discharges (m^3/s)	58	88	116	157	187	219

Table 2-4. Minimum and maximum discharges in Guatapé river, reach downstream of Punchiná reservoir
Source: Integral - Proe S.A.S., 2013

2.4 Climate variability in Colombia

Climate variability along the tropical strip of the Pacific Ocean is caused by the cycles of "La Niña" and "El Niño", variating the winds and the sea surface temperature. "La Niña" represents a cooling phase whereas "El Niño" refers to a heating phase. In Colombia, the ENSO (El Niño–Southern Oscillation) affects largely the Caribbean region. "El Niño" brings low precipitations while "La Niña" is accompanied by heavy rainfall and low temperatures. Both phenomena impact the economy and life of the Colombian people. In the past decade, in the years 2010-2011 "La Niña" affected the region. This event is considered one of the most intense episodes over the last century. On the other hand, "El Niño" affected Colombia in the period 2015-2016 (Fenómeno Niño y Niña - IDEAM, s. f.).

In order to assess the effect of the different meteorological conditions present in Colombia, 3 periods were set up for the multi-objective optimization:

- Simulation period 1: 2010-2011 (La Niña Atmospheric Phenomenon – high precipitation)
- Simulation period 2: 2013-2014 (Averaged weather conditions)
- Simulation period 3: 2015-2016 (El Niño Atmospheric Phenomenon – low precipitation)

2.5 Flood risk at La Pesca village

La Pesca village is located near the confluence of Samaná Norte and Magdalena river, on the left bank of Samaná Norte river mouth (Figure 2-7). Over the last decade, La Pesca village has experienced intense flooding due to high water levels in Samaná Norte river (levels above 126.30 masL). The hydropeaking caused by the operation of the hydroelectric projects in the Samaná Norte watershed is considered one of the causes for this problematic but there are currently no studies to back up this claim. The results from the currently in development PhD dissertation *Implementation of environmental flow regime for multicomponent hydropower generation* and from this study - based solely in the San Carlos Hydroelectric Project - target this query.

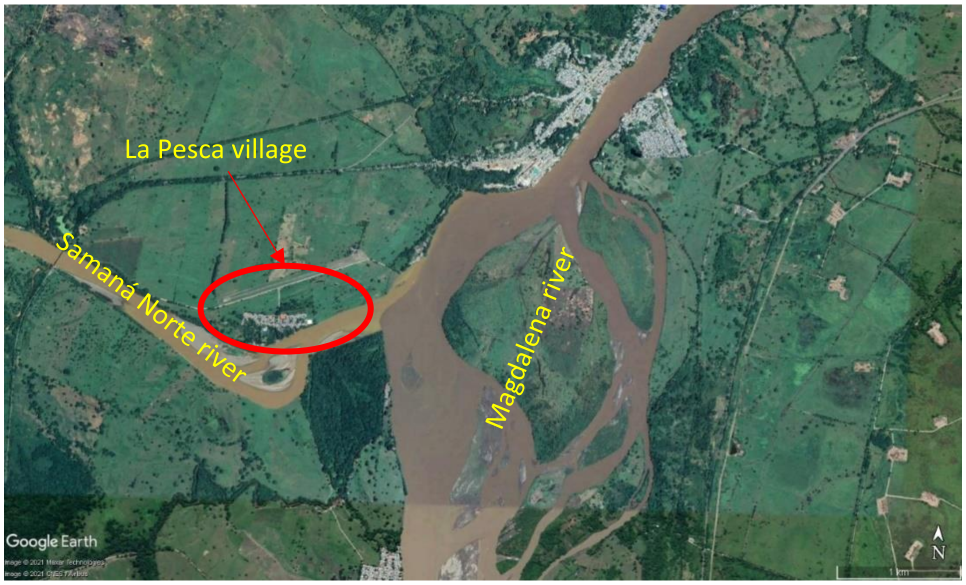


Figure 2-7. Location of La Pesca village
Source: Google Earth

Jairo Villada developed a 1-D hydraulic model of the Samaná Norte river in HEC-RAS for the period 2010-2017 and obtained a rating curve for La Pesca site (Figure 2-8). Due to lack of data, the model has certain limitations. First, for several affluents of Samaná Norte river and for the Magdalena river, the mean monthly discharges per month (m^3/month) were downscaled to daily values assuming each day of the month has the same discharge. This has an important effect on the outcomes of the model since high water levels at Samaná Norte river mouth are highly influenced by high water levels in Magdalena river. Second, a 1-D hydraulic model is not a suitable model for flood assessments since it does not simulate the inundation. However, since the current research and the PhD dissertation have been worked in parallel, the results from this hydraulic model were used.

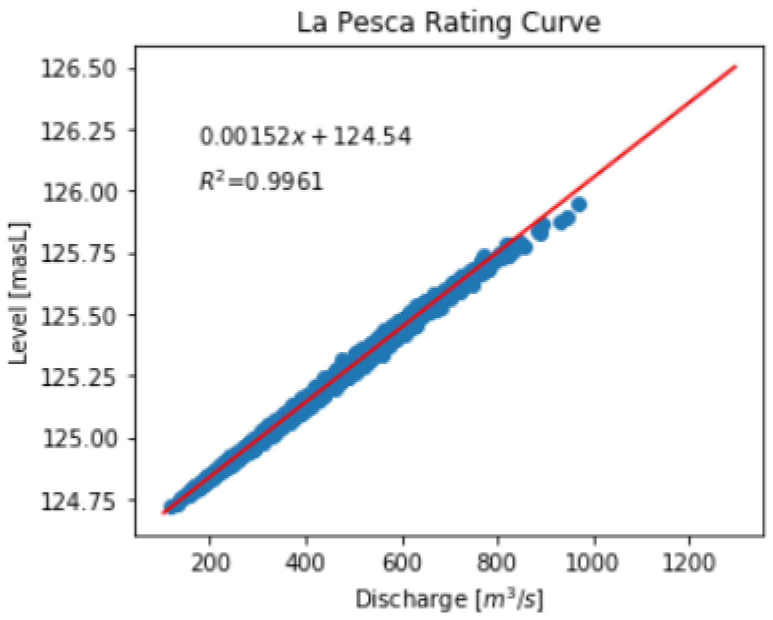


Figure 2-8. Rating curve in Samaná Norte river cross-section at La Pesca village
Source: Villada, 2021

Under the abovementioned model limitations, the model reproduces adequately the water levels at La Pesca village during average flow conditions. During extreme events, there are still some deviations from the actual measurements. A partial conclusion was derived from the current hydraulic model, for floods to occur at La Pesca village 2 situations must convey: simultaneous high discharges at Samaná Norte and Magdalena river.

The Samaná Norte hydraulic model will continue its calibration process after the submission of this report. Two of the improvements will be adding the Magdalena daily discharges (information not available at this moment) and doing a 2D river simulation.

3 Methods

3.1 Reservoir Routing

3.1.1 Definition and numerical methods

Reservoir routing is the procedure to determine the time and magnitude of the outflow hydrograph based on the inflows of the system (rivers, hydroelectric project releases from upstream, rain) and the volume-elevation or surface area-elevation curve of the reservoir. Figure 3-1 shows an example of an inflow and outflow hydrograph within a reservoir. The effect of storage redistributes the inflow hydrograph.

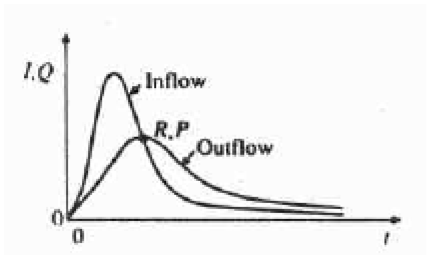


Figure 3-1. Inflow and outflow hydrograph in a reservoir system
Source: Chow et al., 1988

Based on the continuity equation, the rate of change of reservoir storage can be determined as follows:

$$\frac{dS}{dt} = I(t) - Q(t,S) \tag{3}$$

Where S represents the volume of the reservoir [m^3], $I(t)$ is the inflow hydrograph [m^3/s] and $Q(t,S)$ is the outflow hydrograph [m^3/s] dependent of the time and the storage in the impoundment. The outflow can happen through the turbines, if the reservoir is intended for energy generation, a bottom outlet or a spillway.

Equation (3) is a first order differential equation for the storage as a function of time. The change in storage (dS) can also be expressed as a function of the surface area ($A(h)$) and the change in elevation (dh):

$$dS = A(h)dh \tag{4}$$

A new expression for the reservoir can be derived by substituting Equation (4) in Equation (3), resulting in a first order differential equation for the water elevation as a function of time (Chow et al., 1988):

$$\frac{dh}{dt} = \frac{I(t) - Q(t,h)}{A(h)} \tag{5}$$

Numerical methods can be used to solve Equation (5). In this document, 4 methods are applied and coded in Python: Euler, Modified Euler (also known as Runge Kutta 2nd order or predictor/corrector method), Runge Kutta 3rd order and Runge Kutta 4th order.

A higher order method implies higher accuracy; hence, the last method is the preferred one. Nonetheless, if the time step used in the other three numerical methods is quite small, the error reduces significantly and results from different methods do not differ that much. Also, a higher order method implies more computational effort (Fenton, 1992).

- **Euler method**

The simplest method is Euler. It is first order accurate and its formulation is as follows (Fenton, 1992):

$$h_{n+1} = h_n + \Delta F(t_n, h_n) = h_n + \Delta \frac{\left(I(t_n) - Q(t_n, h_n)\right)}{A(h_n)} \tag{6}$$

- **Modified Euler method – Predictor/Corrector method – Runge Kutta 2nd order (RK2)**

Modified Euler is second order accurate two-step method. In this case, the first estimate is determined by Euler's method. Then, the corrector step uses the trapezoidal rule to calculate the value for the following time step (Tenenbaum & Pollard, 1963). The method is shown in Equation (7) while Equations (8) and (9) define the variables K_1 and K_2 respectively.

$$h_{n+1} = h_n + \frac{1}{2}(K_1 + K_2) \tag{7}$$

$$\mathrm{K}_1 = \Delta F(t_n, h_n) = \Delta \frac{\left(I(t_n) - Q(t_n, h_n)\right)}{A(h_n)} \tag{8}$$

$$\mathrm{K}_2 = \Delta F(t_{n+\Delta}, h_n + \mathrm{K}_1) = \Delta \frac{\left(I(t_{n+\Delta}) - Q(t_{n+\Delta}, h_n + \mathrm{K}_1)\right)}{A(h_n + \mathrm{K}_1)} \tag{9}$$

- **Runge Kutta 3rd order (RK3)**

Runge Kutta 3rd order involves the computation of 3 different slopes (K_1/Δ, K_2/Δ and K_3/Δ). Afterwards, a weighted average slope is estimated to compute the solution for the following time step as shown in Equation (10) (Tenenbaum & Pollard, 1963). Equations (11), (12) and (13) define the variables K_1, K_2 and K_3.

$$h_{n+1} = h_n + \frac{1}{6}(K_1 + 4K_2 + K_3) \tag{10}$$

$$\mathrm{K}_1 = \Delta F(t_n, h_n) = \Delta \frac{\left(I(t_n) - Q(t_n, h_n)\right)}{A(h_n)} \tag{11}$$

$$\mathrm{K}_2 = \Delta F\left(t_{n+\Delta/2}, h_n + 0.5\mathrm{K}_1\right) = \Delta \frac{\left(I\left(t_{n+\Delta/2}\right) - Q\left(t_{n+\Delta/2}, h_n + 0.5\mathrm{K}_1\right)\right)}{A(h_n + 0.5\mathrm{K}_1)} \tag{12}$$

$$K_3 = \Delta F(t_{n+\Delta}, h_n - K_1 + 2\mathrm{K}_2) = \Delta \frac{\left(I(t_{n+\Delta}) - Q(t_{n+\Delta}, h_n - K_1 + 2\mathrm{K}_2)\right)}{A(h_n - K_1 + 2\mathrm{K}_2)} \tag{13}$$

- **Runge Kutta 4th order (RK4)**

This represents the most accurate method used along this paper. It is 4th order accurate (Tenenbaum & Pollard, 1963). The estimation of the water level for the next time step is shown in Equation (14) while Equations (15), (16), (17) and (18) define the variables K_1, K_2, K_3 and K_4 respectively.

$$h_{n+1} = h_n + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4) \tag{14}$$

$$\mathrm{K}_1 = \Delta F(t_n, h_n) = \Delta \frac{\left(I(t_n) - Q(t_n, h_n)\right)}{A(h_n)} \tag{15}$$

$$\mathrm{K}_2 = \Delta F\left(t_{n+\Delta/2}, h_n + 0.5\mathrm{K}_1\right) = \Delta \frac{\left(I\left(t_{n+\Delta/2}\right) - Q\left(t_{n+\Delta/2}, h_n + 0.5\mathrm{K}_1\right)\right)}{A(h_n + 0.5\mathrm{K}_1)} \tag{16}$$

$$K_3 = \Delta F\left(t_{n+\Delta/2}, h_n + 0.5\mathrm{K}_2\right) = \Delta \frac{\left(I\left(t_{n+\Delta/2}\right) - Q\left(t_{n+\Delta/2}, h_n + 0.5\mathrm{K}_2\right)\right)}{A(h_n + 0.5\mathrm{K}_2)} \tag{17}$$

$$K_4 = \Delta F(t_{n+\Delta}, h_n + K_3) = \Delta \frac{\left(I(t_{n+\Delta}) - Q(t_{n+\Delta}, h_n + K_3)\right)}{A(h_n + K_3)} \tag{18}$$

3.1.2 Implementation reservoir routing in Python

A python script was developed to simulate the routing process through Punchiná reservoir. The complete code with the example is attached in Appendix 2: Reservoir Routing Code while a detailed description of how to use the program including an example is given in Appendix 3: Description of Python Script. The four numerical methods: Euler, Modified Euler, Runge Kutta 3rd order and Runge Kutta 4th order were coded in Python. Two sets of codes were developed to account for the different types of outflow data (elevation-discharge and time-discharge). Depending on the type of information available, the user must select and run only the corresponding code.

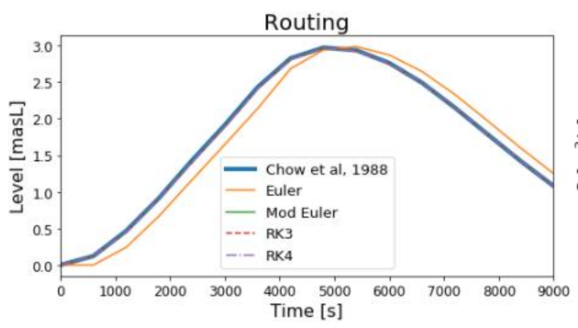
The goal of implementing the four methods was to assess the differences in the routing results and to have the four numerical methods coded in Python for future projects. At the end, to simulate the routing through Punchiná reservoir, the Explicit Euler method was selected because for the available data for this site, it gave better results.

- **Routing example using Python script**

To show that the Python script works properly, the example included by Chow et al. (1988) was run. Both types of outflow data were simulated, and the results were successful (see Figure 3-2 and Figure 3-3). Figure 3-2 (left) shows the water levels inside the reservoir when the outflow data type is elevation-discharge. RK2, RK3 and RK4 return almost the same outcome as given by Chow et al. (1988). Explicit Euler exhibits small differences, underestimating the water levels until 5000 s and then overestimating the reservoir elevations. Most of the times, the differences are below 10%. Figure 3-2 (right) shows the redistribution of the inflow hydrograph caused by the storage of the reservoir. Finally, Figure 3-3 shows the results when the outflow data type is given by time series. The same comments applied for this case.

Given the results, the program was verified. Afterwards, the code was adapted to simulate the routing process through Punchiná reservoir, i.e., including more inflows and outflows.

Outflow data type: Elevation-Discharge Relationship



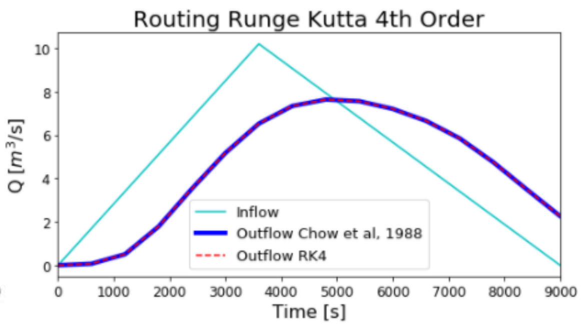


Figure 3-2. Results from routing simulation. Left: reservoir level using the 4 numerical methods. Right: inflow and outflow hydrograph using Runge Kutta 4th Order

Outflow data type: Time series

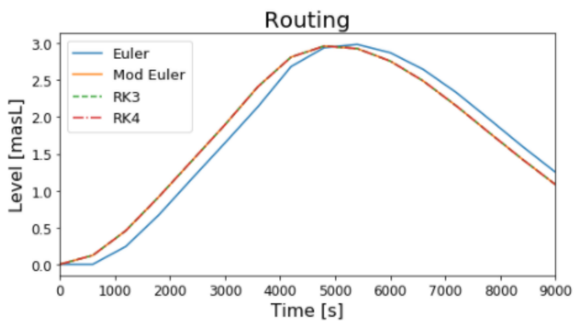


Figure 3-3. Results from routing simulation

3.2 Multi-objective optimization

Reservoir multi-objective optimization problems require a combination of optimization-simulation models because complex nonlinear hydraulic models are difficult to include explicitly as constraints in mathematical optimization methods. Therefore, the control decisions are made by optimizing the objective functions and then the effects on the system are assessed by a simulation model (reservoir routing) (Myo Lin et al., 2020). Figure 3-4 shows the formulation of a multi-objective problem, where f_m are the M objective functions, g_j are the J constraints inequalities, h_k are the K equality constraints, x_i are the N variables to be optimized in the problem, $x_i^L \ and \ x_i^U$ represent the lower and upper bounds for each optimized variable.

$$\begin{aligned}
&\min_x F(x) = \big(f_1(x), \dots, f_M(x)\big) && m = 1, \dots, M, \\
&s.t. \, g_j(x) \leq 0, && j = 1, \dots, J, \\
&h_k(x) = 0, && k = 1, \dots, K, \\
&x_i^L \leq x_i \leq x_i^U, && i = 1, \dots, N
\end{aligned}$$

Figure 3-4. Multi-objective problem definition

Multi-objective optimization problems usually involve the solution of competing objectives. Pareto-optimal solutions are often used as the final product obtained from an optimization problem as these problems always result in trade-offs between the objective functions. Generation of the Pareto-optimal solutions is not an easy task. Multiple algorithms are available to solve these kind of problems (see Figure 3-5). Linear Programming (LP), Non-Linear Programming (NLP) and Dynamic Programming (DP) use a point-by-point approach. These classical methods resort to a weighted or a constraint approach by transforming the multi-objective problem into a single objective optimization, returning only one solution each time. Repetitive optimization runs are required to develop the Pareto front (Deb, 2011). Recently, Evolutionary Algorithms (EA) have gain popularity in solving multi-objective optimization problems because they are easy to use and accessible. The advantage of EA over other methodologies is that it follows a population-based approach, thus, the Pareto optimal front can be found in a single simulation run, i.e., the use of a population allows to find multiple non-dominated solutions simultaneously. Their capacity to solve also multi-objective problems involving non-linearities, non-convexity and large dimensionality have made them widely popular in the recent years (K. Deb et al., 2002). Specifically, Genetic Algorithms have been applied successfully in solving multipurpose reservoirs optimizations (Myo Lin et al. (2020), Malekmohammadi et al. (2011), Reddy & Kumar, (2006)).

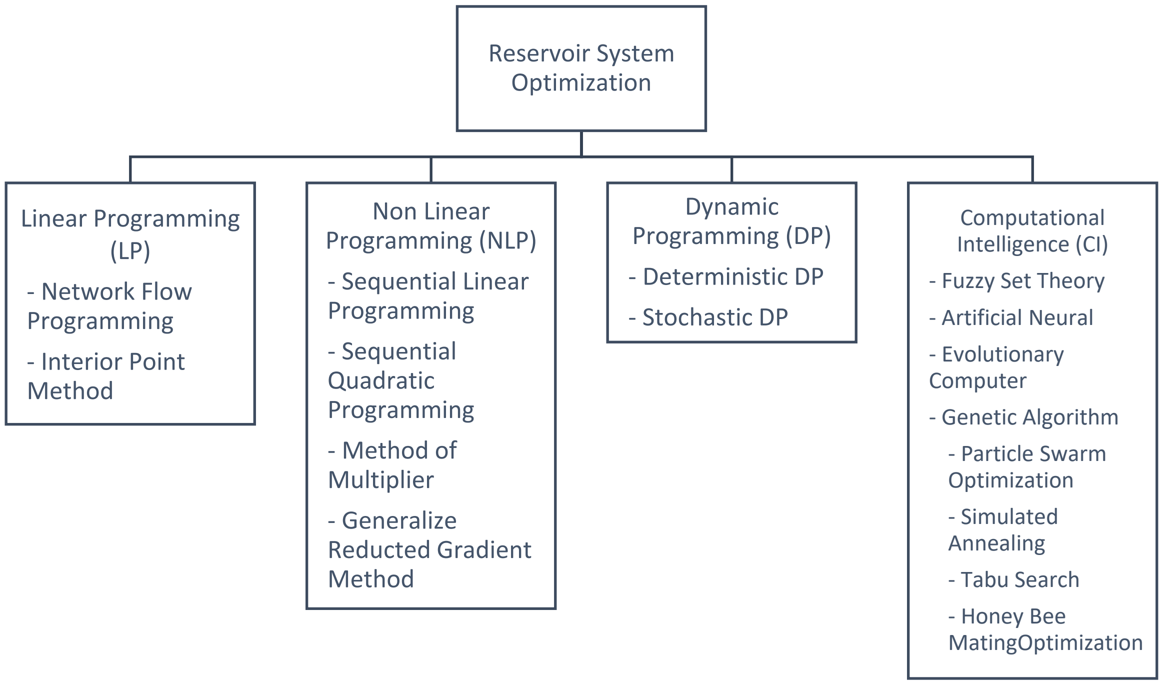

Reservoir System Optimization
Linear Programming (LP)
- Network Flow Programming
- Interior Point Method
Non Linear Programming (NLP)
- Sequential Linear Programming
- Sequential Quadratic Programming
- Method of Multiplier
- Generalize Reducted Gradient Method
Dynamic Programming (DP)
- Deterministic DP
- Stochastic DP
Computational Intelligence (CI)
- Fuzzy Set Theory
- Artificial Neural
- Evolutionary Computer
- Genetic Algorithm
- Particle Swarm Optimization
- Simulated Annealing
- Tabu Search
- Honey Bee MatingOptimization

Figure 3-5. Algorithms used for reservoir optimization
Source: Ahmad et al., 2014

3.2.1 Genetic Algorithm

"Genetic Algorithms are the population based search and optimization technique that mimic the process of natural evolution" (Kora & Yadlapalli, 2017). The performance of the evolutionary algorithms depends on genetic operators:

- Population sampling: at the beginning, an initial population is set based on sampling.
- Crossover: is associated with the mating process. It represents the combination of genetic information from two parents to generate a new offspring (Kora & Yadlapalli, 2017).
- Offspring mutation: once the offspring are created, the mutation is carried out using a preselected probability. This allows for more diversity within the population (Blank, 2020).
- Elite-preservation: combines the old population with the new population and only keeps the better solutions from the combined population (Kalyanmoy Deb, 2011).

The procedure for solving an optimization problem using a genetic algorithm is explained in this paragraph. First, an initial population value is fixed (the size is defined by the user and is dependent of the size of the problem). Then, the population is evaluated using the defined problem. The fittest solutions survived. Afterwards, the mating process starts. Initially, there is a selection to define the parents for the next generation. The crossover helps to define how the chromosomes of the parents are combined and the mutation brings more diversity into the next generation. The main idea of the genetic algorithm is to produce an improve offspring (Kora & Yadlapalli, 2017). Then, this new generation is evaluated and the process enters a loop until the optimal solution is found (Blank, 2020).

The GA procedure uses the objective information directly, i.e., does not use gradient information to find the solutions. Moreover, the stochastic operators use within the method allow to overcome the problems of local optima and other complexities (Kalyanmoy Deb, 2011).

K. Deb et al. (2002) developed a fast and elitist multi-objective genetic algorithm named non dominated sorting genetic algorithm II (NSGA-II). The improved methods displayed a better performance than the Pareto-archived evolution strategy (PAES) and the strength Pareto EA (SPEA).

In this report, the *Pymoo* framework in Python was selected to address the single-objective and the multi-objective optimization by applying Genetic Algorithm (GA) and NSGA-II, respectively. *Pymoo* considers only minimization problems. A maximizing objective can be included by multiplying the objective function by -1. Moreover, the constraints must be entered as a less-than-equal-to constraint. Therefore, if a higher-than-equal-to constraint must be inputted, the equation must be modified to follow the upper statement. To illustrate this point, Equation (19) represents the problem constraint and Equation (20) is how this constraint must be inputted when using *Pymoo*.

$$x \geq -2 \tag{19}$$

$$-x - 2 \leq 0 \tag{20}$$

In case equality constraints are required (as exemplified in Equation (19)), these are handled in *Pymoo* by expressing them as inequality constraints smoothed by adding an epsilon as shown in Equation (22). By writing the equality constraints in this manner, strict constraints that make the search space infeasible are avoided (Blank & Deb, 2020).

$$x = 5 \tag{21}$$

$$g'(x): (x-5)^2 - \hat{\epsilon} \leq 0 \tag{22}$$

3.3 Multi-objective optimization at Punchiná reservoir

The target of the multi-objective optimization at Punchiná reservoir is to allow high revenues from hydroelectric generation while reducing the flood hazards downstream at La Pesca Village and allowing flow throughout the year in the Guatapé river reach, downstream of the dam. To solve the multi-objective optimization, the non-dominated sorting genetic algorithm II (NSGA-II) using the *Pymoo* framework in Python was set up, along with the use of an Explicit Euler numerical method for modelling the river routing (see Python code in Appendix 4: Multi-Objective Optimization Code).

As explained before, the inflows into Punchiná reservoir are the San Carlos river discharge, the spill and turbined discharge from Las Playas Power Plant. Currently, the only outflows of the system are the turbined and the spill discharges. The latter flows along the Guatapé river while the turbined flow discharges directly into Samaná Norte river. To have flow all-year-long in Guatapé river, downstream of Punchiná dam, an outlet discharge is introduced. The possibility of using the bottom outlet to release flow is discarded since is prohibited its operation in Colombia. Furthermore, the quality of the water released by bottom outlets is not considered suitable for the species downstream of the dams because of the turbidity and temperature. Therefore, this outlet outflow requires setting a pumping station or a structure that allows flow releases every day into Guatapé river. A scheme of the system is shown in Figure 3-6.

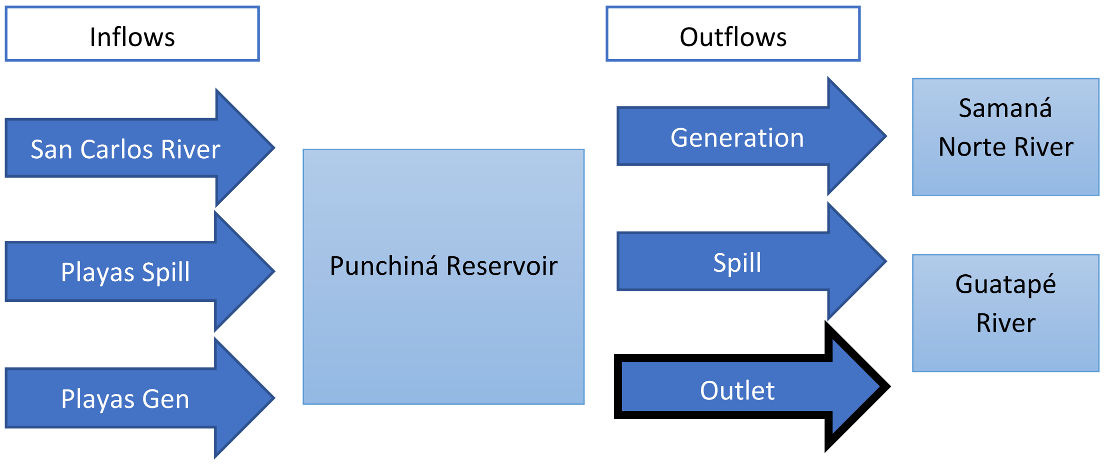


Figure 3-6. Punchiná system operation for multi-objective optimization

An optimization problem (Figure 3-4) requires the definition of the optimization variables, objective functions and constraints – for the optimization variables and the optimization problem.

3.3.1 Optimization variables

The variables that are being optimized are:

- $Q_{Gen,t}$: turbined flow at San Carlos plant
- $Q_{Outlet,t}$: outlet flow that discharges into Guatapé river

3.3.2 Constraints for the optimization variables

- **Generation Discharge**

These constraints are dependent on the equipment displayed in the Powerhouse. San Carlos powerhouse has 8 turbines, each with a net effective capacity of 155 MW – 1240 MW in total. By dividing the net capacity by the Conversion Factor, the minimum (one turbine working) and maximum (all turbines working) discharges are calculated and inputted as constraints in the model. Equation (23) shows the lower and upper bound of the generation discharges.

$$29.5\ m^3/s\ \leq Q_{gen,t} \leq 226.9\ m^3/s \tag{23}$$

- **Outlet Discharge**

Currently, at Guatapé river, reach downstream of Punchiná dam, the average flow throughout the year is 8.3 m^3/s and it is bound to flows over San Carlos spillway. Thus, on daily basis, there is no flow at this reach. Since there is no data available regarding the Guatapé river before the human interventions in the watershed, a range between 20 and 30 m^3/s is set based on the fact that if the turbines could discharge into this river, the minimum flow they could delivered is 29.5 m^3/s per turbine.

After running the models, corrections were necessary for the low discharge years (“El Niño” period). No feasible solutions arose given the available flow; thus, the outlet discharge range was reduced. The final constraints are summarized in Equation (24).

$$\begin{gathered} 20\frac{m^3}{s} \leq Q_{Outlet,t} \leq 30\frac{m^3}{s}\ for\ Simulation\ period\ 1\ and\ 2 \\ 10\frac{m^3}{s} \leq Q_{Outlet,t} \leq 30\frac{m^3}{s}\ for\ Simulation\ period\ 3 \end{gathered} \tag{24}$$

The above values represent already an improvement from the current situation (Table 2-3). A range was selected for the ecological flow instead of a steady flow because the European Commission (2014) said that natural flow regimes present variability at different time scales - seasonally and inter annually – and native aquatic and riparian biota adapt to this variability. Thus, the magnitude, frequency, duration, timing, and rate of change of the natural flow is the key to sustaining and conserving native species and ecological integrity.

3.3.3 Objective functions

- Objective 1: Maximizing Revenues from Energy Generated by Hydropower

The main objective in a hydroelectric project is to generate the higher amount of revenues to make the project feasible. Under this frame, the first objective was defined as:

$$f_1 := \sum_{t=1}^{T_{end}} Q_{Gen,t} * FC(H_t) * Price_t * 1000 * 24 \quad [\$] \tag{25}$$

Where $Q_{Gen,t}$ is the generated outflow at time t in m^3/s, H_t is the reservoir level at time t in masL, $Price_t$ is the energy price at time t in \$/kWh (already in NPV to reduce the computation time), T_{end} is the number of time steps, t is the time in days, 24 converts the discharge from MW/day to MWh, FC is the conversion factor equation used to transform the discharge from m^3/s to MW/d and 1000 transforms the MW to kW. At the end, f1 has units of Colombian peso [\$].

Depending on the time of the year and the climatological conditions, energy price fluctuations arise. Figure 3-7 shows the daily and monthly energy price fluctuations of the energy price in units of \$/kWh. The period 2010-2011 was affected by "La Niña", the period 2015-2016 was influenced by "El Niño" and the other years represent average meteorological conditions in Colombia. As a consequence of "El Niño", the prices were higher in the years 2015 and 2016. For the other years (2010-2014, 2017), the price fluctuation display a similar trend.

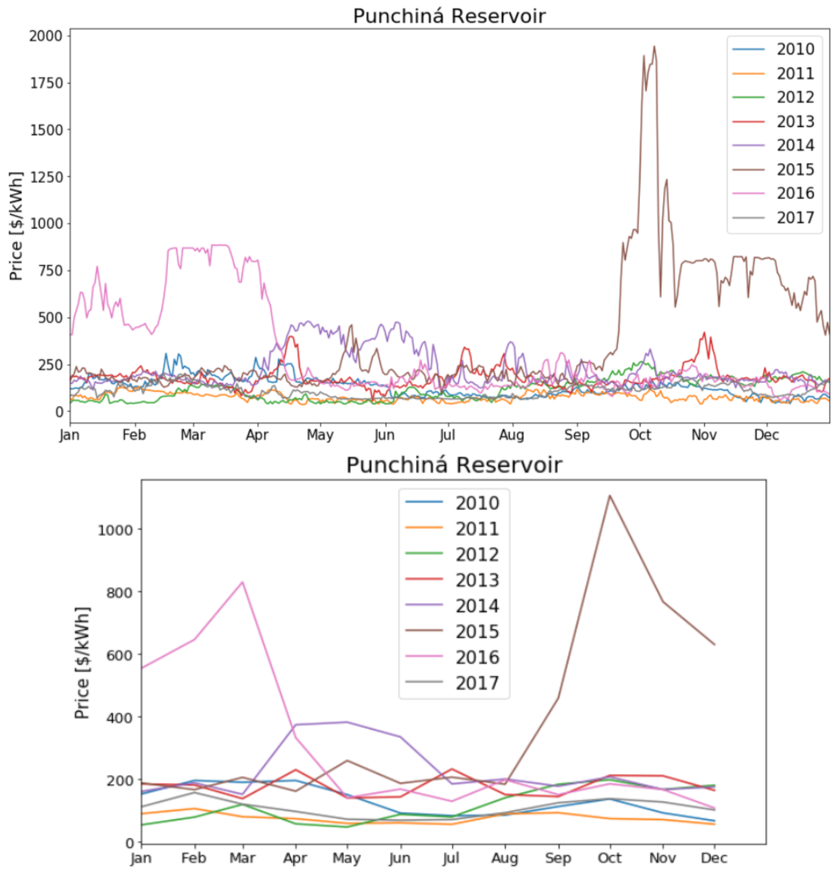


Figure 3-7. Energy prices in units of \$/kWh in the period 2010-2017. Top image: daily. Bottom image: monthly

Source: (XM, s. f.)

The prices were transformed to Net Present Value (see Equation (26)) set at the beginning of each simulation period using the monthly nominal discount rates from Banco de la República de Colombia (see Appendix 1: Case Study Data).

$$NPV = \frac{FV}{\left(1 + \frac{i}{100}\right)^n} \tag{26}$$

Where NPV is the net present value of cash flow in \$, FV net cash inflow-outflow during a single period n in \$, i is the nominal discount rate in % and n represent the number of periods.

The incorporation of the price affects largely the optimization problem. The goal is not to generate the more amount of energy per year but to generate the more amount of revenues, i.e., to turbine more flow during the months where the prices are higher.

- Objective 2: Maximizing the ecological discharge at Guatapé river

Hydroelectric projects have severe environmental impacts. Downstream of a dam the natural flow regime is largely modified. At the moment the San Carlos Project was built, no ecological discharge considerations were taken into account for the Guatapé river, leaving a large part of the reach in dry almost all year long with flow only during spill days. To rectify this situation, a maximizing objective is set so that the sum of spill flow and outlet flow (Equation (27)) ensures a continuous flow all year long.

$$f_2 := \sum_{t=1}^{T_{end}} Q_{Spill,t} + Q_{Outlet,t} \quad \left[\frac{\mathrm{m}^3}{\mathrm{s}}\right] \tag{27}$$

Where $Q_{spill,t}$ is the spill discharge at time t in m^3/s and $Q_{Outlet,t}$ is the outlet discharge at t in m^3/s.

- Objective 3: Minimizing the flood risk at La Pesca village

The aim of this objective is to mitigate the flood risk at La Pesca village by controlling the turbined discharges from San Carlos Hydroelectric Project through the penstock. Since preventing floods may not always be possible and may not lead to feasible solutions, a variable called exceedance level was introduced (Lugt, 2018). The objective is defined then to minimize the exceedance water levels in the Samaná Norte river, at the site of La Pesca (Figure 3-8) as shown in Equations (28) and (29).

$$f_3 := \sum_{t=1}^{T_{end}} Level_{exc,t} \quad [m] \tag{28}$$

$$Level_{exc,t} = \begin{cases} 0 \; if \; Level_{La\ Pesca,t} \leq Level_{warning} \\ Level_{La\ Pesca,t} - Level_{warning} \; if \; Level_{La\ Pesca,t} > Level_{warning} \end{cases} \tag{29}$$

Where $Level_{exc,t}$ is the exceedance water level at La Pesca village in masL, $Level_{La\ Pesca,t}$ is the water level at La Pesca site in masL obtained from the rating curve displayed in Figure 2-8 (where discharge refers to the discharge at Samaná Norte river at the river mouth) and $Level_{warning}$ is a warning level set at 126.00 masL as an alert system.

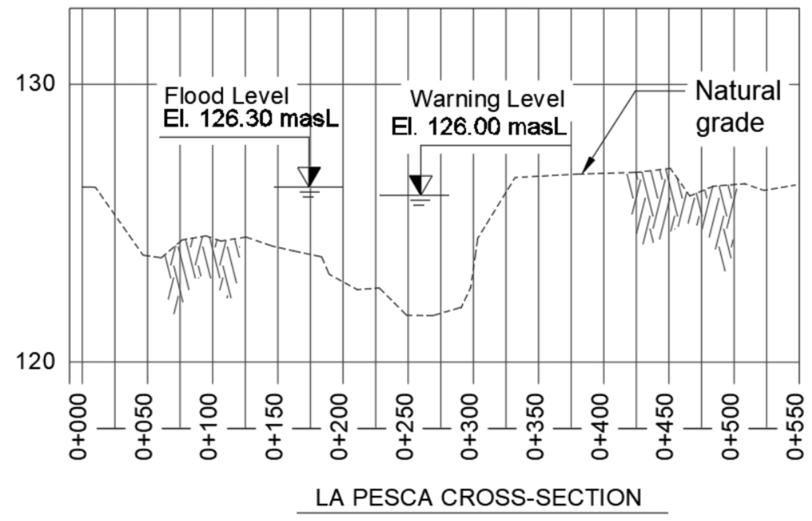


Figure 3-8. Samaná Norte river cross-section at La Pesca village

For each time step, the discharge at Samaná Norte river mouth (Equation (31)) is estimated by adding the $Q_{San\ Carlos\ Project\ discharges}$ (Equation (30)) to the generated timeseries. Afterwards, the water level at La Pesca is interpolated from the rating curve displayed in Figure 2-8.

$$Q_{San\ Carlos\ Project\ discharges,t} = Q_{Gen,t} + Q_{Outlet,t} + Q_{spill,t} \tag{30}$$

$$Q_{Samaná\ river\ mouth,t} = Q_{San\ Carlos\ Project\ discharges,t} + Q_{SRV,t} \tag{31}$$

Where Q_{SRV} is the discharge at Samaná Norte river mouth excluding the flow from San Carlos Hydroelectric Project in m^3/s.

No flow routing is performed along the Samaná Norte river since the propagation time of the wave is less than a day.

3.3.4 Constraints for the optimization problem

The optimization also requires inequality constraints to be written in the form: $g_n\ (X) \leq 0$. Hence, four constraints were defined to ensure the system and the optimization problem behave as expected.

- **Constraints for the water levels in the reservoir**

The operational water levels in the reservoir are defined from 754 masL -MOL- until 781 masL -dam crest level. At low reservoir levels, when the water levels are close to the MOL, an inlet vortex can be formed, allowing air to be sucked into the system. This is problematic because the efficiency of the turbine is reduced because the air bubbles occupy part of the volume inside the waterway and the bubbles may explode when reaching the turbine, causing damages to the blades and reducing the energy generation. The constraints for the water level are therefore defined as:

$$754\ masL \leq WL_t \leq 781\ masL \tag{32}$$

$$g_1 = WL_t - 781 \leq 0 \tag{33}$$

$$g_2 = -WL_t + 754 \leq 0 \tag{34}$$

Where WL_t is the water level inside the reservoir in masL, g_1 represents the first inequality constraint related to the maximum safe water level inside the reservoir and g_2 is the second inequality constraint related to the minimum operational water level inside the reservoir.

- **Constraints for the discharge through the spillway**

The spillway is designed for an instant maximum discharge capacity of 7200 m^3/s. As the optimization is performed on daily basis, the maximum discharge that can flow over the spillway was defined as the maximum volume that needed to be evacuated in a single day ($Vol_{H=781} - Vol_{H=775}$) divided by 86400 s. Therefore, the constraints for the discharge through the spillway are defined as:

$$Q_{spill,t} \leq Q_{max\ spill} \tag{35}$$

$$Q_{spill,t} - Q_{max\ spill,day} \leq 0 \tag{36}$$

$$g_3 = Q_{spill,t} - 231.45\ m^3/s \leq 0 \tag{37}$$

Where $Q_{max\ spill}$ is the maximum discharge that can be evacuated in a single day in m^3/s and g_3 represents the third inequality constraint related to the discharge through the spillway.

- **Constraints for the water levels at La Pesca village**

The inequality constraint that allows to penalize the exceedance flood level instead of limiting the water level (Lugt, 2018) is given as:

$$g_4 = Level_{La\ Pesca,t} - Level_{exc,t} - Level_{warning} \leq 0 \tag{38}$$

Where g_4 represents the fourth inequality constraint of the multi-optimization problem.

- **Constraints for the initial and final water levels**

To find a rule curve, the initial and final water levels inside the reservoir must have the same value. Since the optimization is performed on daily basis, the constraint requires that the water level on January 1st must be equal to the water level on December 31st . Following the equality constraint handling explained in Chapter 3.2, this constraint is defined as:

$$g_5 = (WL_{t=1} - WL_{t=365})^2 - \hat{\epsilon} \leq 0,\ \ \hat{\epsilon} = 0.001 \tag{39}$$

Where $WL_{t=1}$ is the water level on January 1st in masL, $WL_{t=365}$ is the water level on December 31st in masL, $\hat{\epsilon}$ is added to smooth the constraint and g_5 represents the fifth constraint of the multi-optimization problem.

3.3.5 Routing

Inside the optimization problem, the reservoir routing is being estimated since water level and discharge are fully related, and both need to be known for each time step. Due to the computational effort and time required to solve the optimization problem, the Explicit Euler method is being used to solve the first order differential equation for the water elevation as a function of time .

3.3.6 Spillway operation

Inside the routing, the spill discharge is estimated for each time step. The spill discharge only happens when the water level inside the Punchiná reservoir exceeds the crest spillway level (775 masL).

$$Q_{spill,t} = \begin{cases} 0 \; if \; WL_t \leq Level_{spillway\ crest} \\ (Vol_{H>775} - Vol_{H=775})/86400 \text{ if } WL_t > Level_{spillway\ crest} \end{cases} \tag{40}$$

As stated by Lugt (2018), if the flow through the spillway would be a free variable in the optimization problem, an optimal solution for maximizing ecological flow would be to spill a lot of water when the reservoir levels are below the crest spillway level. Therefore, by defining the spill discharge as a result of the routing and by following the abovementioned constraints for water levels inside the reservoir, the correct formulation for the optimization problem was obtained.

4 Results and discussion

4.1 Punchiná reservoir routing

Before solving the multi-objective optimization, the reservoir routing was carried out to estimate the discharges through the spillway of Punchiná's reservoir in the period 2010-2017. This step was needed because the measurements in this period were not completely reliable. Spill discharges of 0 m^3/s were recorded most of the days within this period; nonetheless, given the inflows (San Carlos river discharge, discharges from the generation in Playas Hydroelectric Plant and spillway discharges from Playas dam), outflows (generation discharges in San Carlos Hydroelectric Project and spillway discharges from Punchina's dam) and initial conditions of the reservoir (water level at time 0), this implied that the water elevation inside the reservoir would had reach elevations higher than physically possible (higher than the crest of the dam).

Given the above situation, the quality of the measured information was questioned. Therefore, a thorough research of the different routing inputs was conducted to understand the abovementioned behavior and to be able to provide reliable reservoir routing outcomes. In the following paragraphs this research is explained in detail to show the required data evaluation and correction performed in this report.

First, the reservoir capacity curve was further analyzed. However, after thorough investigation, it was confirmed that the Volume vs Elevation curve and Area vs Elevation curve of the Punchiná reservoir were correct.

Second, the discharges from the generation in Playas Hydroelectric were checked. At first, a constant conversion factor was used since the water elevation at Playas reservoir was not available. Using a constant conversion factor is not adequate because when transforming the generation into discharges, the influence of the water level within the reservoir is neglected. This effect must be considered since the efficiency of the turbines varies with the head. More accurate discharges from the generation in Playas Hydroelectric Plant were obtained based on the reservoir curve from Díaz (2011), the time series of volume inside Playas reservoir (XM, s. f.) and the Las Playas conversion factor equation. Appendix 5: Inflow discharges at San Carlos Project shows the inflows for the years 2010-2011, 2013-2014 and 2015-2016.

Third, the generation discharges of San Carlos Hydroelectric Plant were reviewed. The approach explained in the previous paragraph was applied. For this plant, the capacity curve of the reservoir was already provided by the electric company.

After reviewing the reservoir capacity curve and the generated discharges from Playas and San Carlos projects, the routing still returned water levels higher than physically possible. Since the problem persisted, it was considered that the conversion factor equations (Equations (1) and (2)) are not well established; hence, the generation discharges from both projects, estimated in this document, may deviate from the real values. Several attempts were made to calibrate the conversion factors of both projects with no success.

Finally, it was observed that the discharges through the spillway of San Carlos Power Plant were not properly measured. This conclusion was reached after looking closely at the provided data. Many days, the water levels measured at Punchiná were below the spillway crest level; however, spill discharges were monitored. On the other hand, when the water levels where above the crest level, often no spill discharges were registered. Hence, the measurements concerning water levels and spill discharges were not congruent.

Given the fact that the data series of spill discharges at Punchiná were not reliable and that the routing returned higher water levels than physically possible when using the measured data, it was decided to correct the spillway discharges by estimating them directly from the routing, i.e., when the routing returned water levels higher than the spillway crest level, the spill discharge was computed and used as an input for the following time step.

The routing was performed in 2 stages. First the period 2010-2011 and then the period 2012-2017. The difference is that for the period 2010-2011, interpolation from the reservoir curve capacity was done to estimate the water levels inside the reservoir. Therefore, the first initial condition (water elevation at time 0) came from the interpolated value. For the second period, the water levels were monitored, hence, the water level at time 0 came from a measured value.

Figure 4-1 and Figure 4-2 show the routing results for both periods. For the years 2010-2011 and 2016-2017 the routing water levels showed good agreement with the measured water levels. The period 2012-2016 shows a large difference between the routing and the recorded levels (Figure 4-1). Though, for this 5-year period no spill flows were monitored (Figure 4-2). When contrasting the spills discharges from measured and routing process in Figure 4-2, the routing reproduces fairly good all measured overflows, however, the routing also shows that in many days there should have been overflows to keep the system working. These results allow to conclude that there is a monitoring problem in the San Carlos project.

The routing results from Figure 4-1 and Figure 4-2 were used in the optimization problem as the measured scenario.

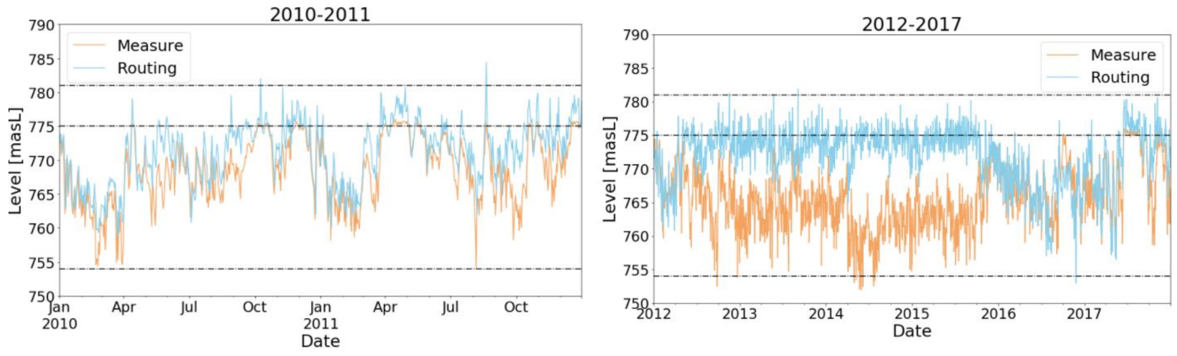


Figure 4-1. Water levels measured and computed at Punchiná reservoir. Left: 2010-2011. Right: 2012-2017

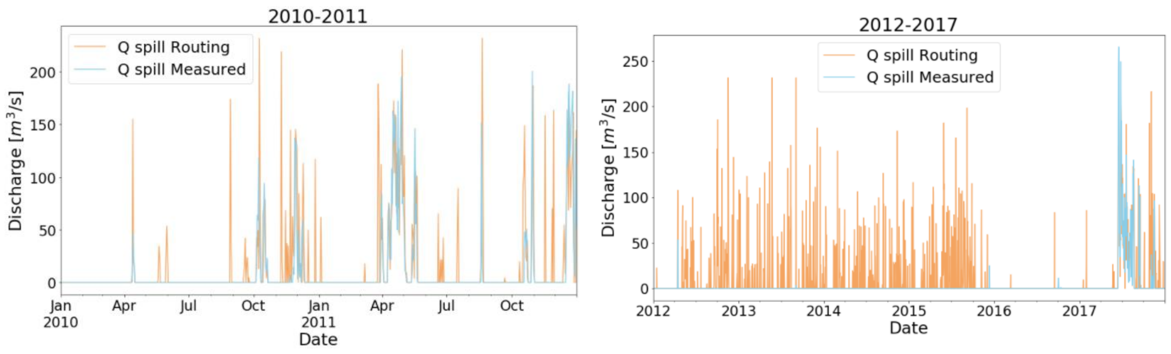


Figure 4-2. Spill discharges measured and computed at Punchiná reservoir. Left: 2010-2011. Right: 2012-2017

4.2 Optimization

Deriving monthly operation curves for the operators of San Carlos Hydroelectric Plant was the goal of the multi-objective optimization. These curves are the result of averaging the daily optimization outcomes. To provide operation curves for the different meteorological conditions present in Colombia within the 2010-2017 interval, 3 simulation periods were set up for the multi-objective optimization: 2010-2011 (La Niña Atmospheric Phenomenon – high precipitation), 2013-2014 (Averaged weather conditions) and 2015-2016 (El Niño Atmospheric Phenomenon – low precipitation).

The optimization uses as input data the inflows into the system averaged over a year period, i.e., although each simulation period consists of 2 years, the simulation timeframe was set to a year (Figure 4-3).

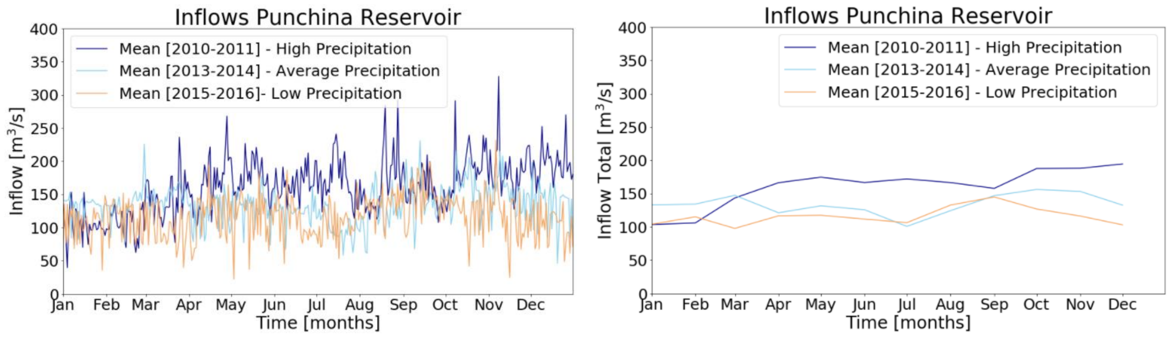


Figure 4-3. Average i*nflow discharges at Punchiná reservoir for the 3 simulation periods Left: daily. Right: monthly*

To assess the trade-offs between the 3 objectives – maximizing revenues, maximizing ecological flow and minimizing flood risk downstream – the following 3 MO optimization scenarios were simulated:

- Scenario 1: a single-objective optimization based solely on maximizing revenues.
- Scenario 2: a bi-objective optimization was solved by adding the ecological discharge objective.

- Scenario 3: the flood mitigation objective was added in the multi-objective optimization that includes all three objectives. This scenario was simulated only in the 2010-2011 period due to the high discharges present along the Samaná Norte system.

Scenario 1: Maximizing hydropower revenues

When only the revenues are maximized, the spill discharges reduce to zero in the 3 simulation periods, letting no flow at Guatapé river - downstream of Punchiná dam (see Table 4-1). The rule is that the operator must let the flow go through the turbines all-year-long and avoid flow losses over the spillway. When contrasting against the registered situation (see Table 4-1), spills occurred during the 3 analyzed periods letting average daily discharges in Guatapé river of 14 m^3/s, 7 m^3/s and 12 m^3/s, respectively. Nonetheless, as explained before and as shown in Figure 4-2, the spill flows happened sporadically and usually involve high discharges. Thus, it is noted how the single objective optimization impacts the environment downstream by drying the Guatapé river completely. Moreover, it releases high discharges during wet periods directly into Samaná Norte river, which may lead to floods downstream. Operating the reservoir only with this aim endangers the ecosystem services by damaging the natural environment and endangering the people downstream. Nevertheless, the operation over the past 3 decades has strived for this objective.

Table 4-1 also shows how in all cases, the single-objective optimization returned around 10% higher annual energy generation, i.e., it was possible to exploit even more the project.

For the different meteorological conditions, the simulation shows that the reservoir levels can be kept quite constant during the year (Figure 4-4). For the high discharges period (La Niña), the average monthly flows reach 200 m^3/s while for low discharges period (El Niño), the average monthly flows are around 125 m^3/s.

Period	Solution	Revenues [$] (1)	Annual Energy [GWh]	Average Daily Discharge in Guatapé River [m^3/s]
2010-2011	Measured	6.41E+11	6998.5	14.04
	Optimization	7.15E+11	7756.1	0
2013-2014	Measured	9.98E+11	5191.6	7.36
	Optimization	1.09E+12	5684.5	0
2015-2016	Measured	8.25E+11	5721.3	12.02
	Optimization	9.41E+11	6489.7	0

(1) Prices in Colombian Peso ($)

Table 4-1. Multi-objective optimization results for the period 2010-2011

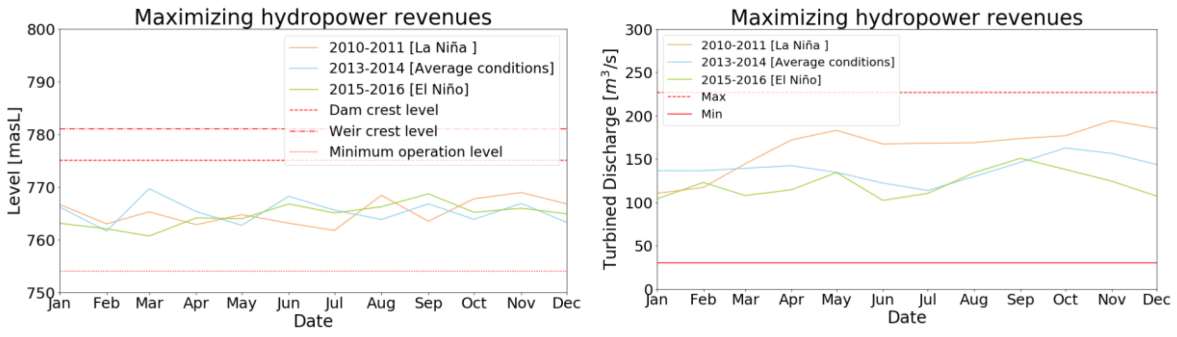


Figure 4-4. Monthly single objective results, 3 simulation periods. Left: operation curve. Right: turbined discharges

Scenario 2: Maximizing hydropower revenues and maximizing ecological discharge via a Bi-objective optimization

From the previous results, an outcome of maximizing hydropower revenues only was zero flows over the spillway, i.e., no flow in Guatapé river – reach downstream of Punchiná dam. By incorporating the second objective in the optimization problem, compromising solutions between the 2 objectives arose.

Figure 4-5 shows the Pareto Optimal Solutions when the outlet discharge is constraint according to Equation (24). The 2 competitive objectives reveal that when the revenues are the highest possible within the problem set-up, the ecological discharge is the lowest possible at Guatapé river and vice versa. To better visualize these trade-offs in the operation of the reservoir, Figure 4-6 displays the reservoir levels and discharges for the 3 simulation periods, for the solutions where each objective has its maximum value.

When Objective 1 is having more weight in the optimization problem, the reservoir levels are kept usually below the spillway crest level -implying small quantities of spill flow along the year, the turbined discharges are high and the Guatapé river discharges are quite constant and bound to the outlet flow range.

An interesting outcome is that for Simulation periods 2 and 3 (average and low discharge conditions) the average turbined flows are quite similar. Nonetheless, the water levels inside the reservoir are kept higher for the second case in order to withstand periods of lower discharges.

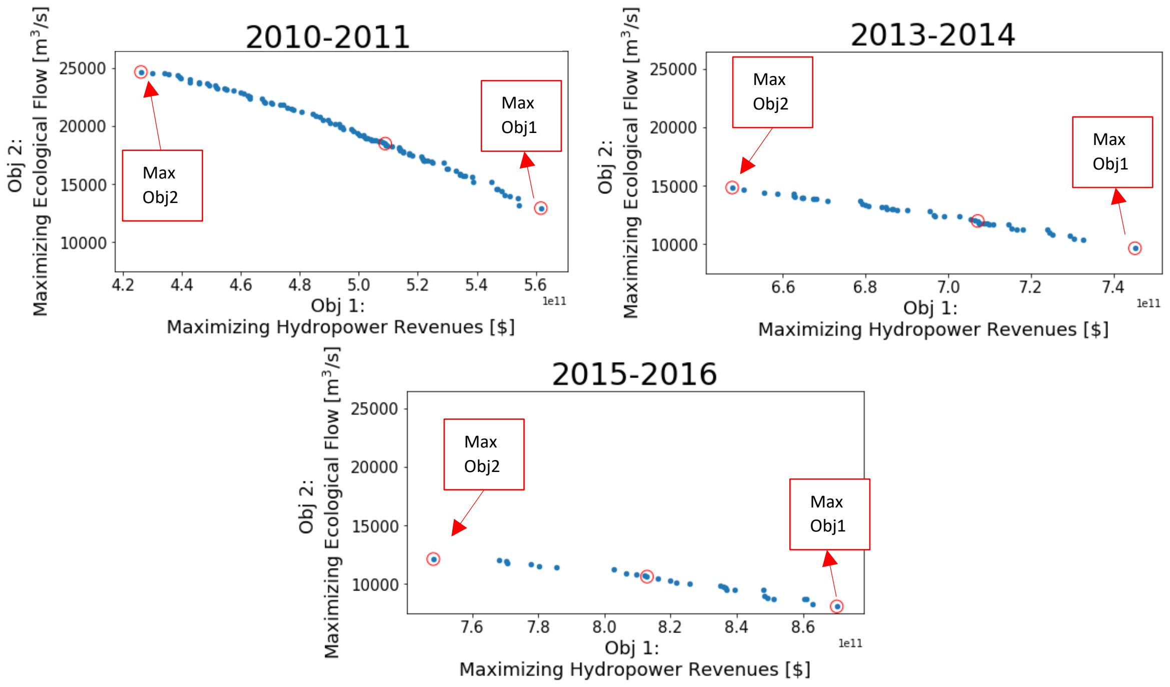


Figure 4-5. Pareto Optimal Solutions for bi-objective optimization, outlet discharge in the range 20-30 m^3/s for the simulation period 1 and 2 and 10-30 m^3/s for the simulation period 3. Upper Left: period 2010-2011. Upper Right: period 2013-2014. Bottom: period 2015-2016.

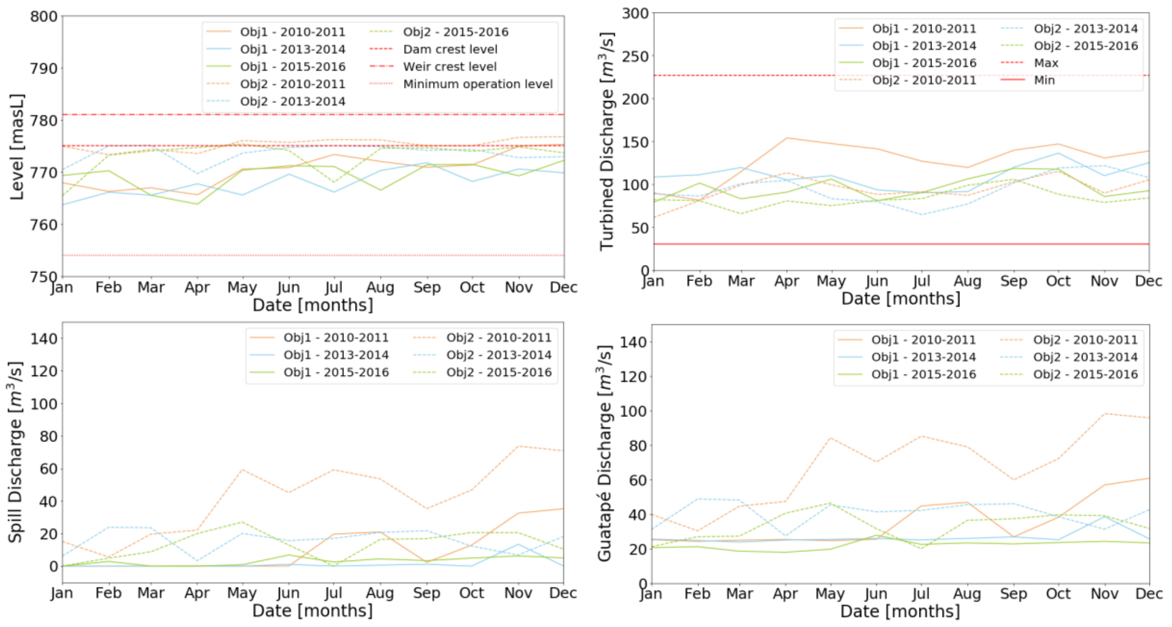


Figure 4-6. Monthly bi-objective results, 3 simulation periods. Upper Left: operation curve. Upper Right: turbined discharges. Lower left: spill discharges. Lower right: Guatapé river discharge

Table 4-2, Table 4-3 and Table 4-4 summarizes the results from the single and bi-objective optimization and compares the simulation against the registered situation for the 3 simulation periods. Appendix 6: Multi-objective optimization results. Period 2010-2011, Appendix 7: Multi-objective optimization results. Period 2013-2014 and Appendix 8: Multi-objective optimization results. Period 2015-2016 exhibits the outcomes graphically.

The tables show the trade-offs between hydropower revenues and ecological discharge while operating San Carlos Hydroelectric Plant. When contrasting the solutions where the revenues are maximized for the single and bi-objective, the inclusion of an outlet flow impacts the revenues and annual energy by 20% approximately, for the 3 simulation periods. The operation rules and the turbined flows (Figure 4-4 and Figure 4-6) change to suffice a continuous flow in Guatapé river. If more importance is given to the ecological flow objective, the impacts on the yearly revenues become even higher. To quantify monetarily its effect, an ecological discharge price was set, assuming this flow has the same values as the turbined flow.

To assess other outlet discharge ranges, another bi-objective optimization was run for the simulation period 2010-2011. For this case, the lower bound of the outlet discharge range was set at 5 m^3/s and the upper bound at 15 m^3/s. When Objective 2 becomes the main purpose of the optimization, the ecological discharge price is practically the same for both cases. Therefore, for this period it is the maximum yearly price that the ecological discharge may have. This procedure can be carried out for other simulation periods to see if a maximum price can be set for every meteorological condition.

Table 4-2 also attempts to establish the outlet discharge that can be taken from Punchiná reservoir without affecting the hydropower generation. By reducing the outlet discharge and by setting the Objective 1 as the main target, a 16.5% decrease in energy generation and a 15% decrease in revenues is experienced. This operation rule looks appealing since the loss of revenues and the price of the ecological discharge is the lowest while at the same time it provides an average flow of 27 m^3/s in the Guatapé river, reach downstream of Punchiná dam.

Although letting an ecological flow at Guatapé river throughout the year always results in revenues reduction, there is an ecosystem services gain. The improvement of these services are difficult to quantify but impacts positively the society and the environment. Even for a period where El Niño affects Colombia, mean discharges of 20 m^3/s can be delivered to the river reach. Therefore, even if the natural flows are low for a given year, there is enough resource to suffices both objectives by making compromises in the generation and environmental sector. In the following Chapter recommendations are given to motivate the electric companies to commit to these compromises.

Optimization Type	**Outlet discharge range [m^3/s]**	**Solution**	**Revenues [$] (1)**	**Revenues reduction [%] (2)**	**Annual Energy [GWh]**	**Annual energy reduction [%] (2)**	**Average Daily Discharge in Guatapé River [m^3/s]**	**Ecological discharge price [$]**
[-]	[-]	Measured	6.41E+11	[-]	6998.5	[-]	14.04	[-]
Single Objective	[-]	Max Hydropower	7.08E+11	[-]	7648.4	[-]	0.00	[-]
Bi-Objective	[20-30]	Max Hydropower	5.62E+11	20.6	6030.5	21.2	35.44	1.46E+11
		Max Ecological	4.26E+11	39.8	4492.6	41.3	67.51	2.81E+11
		Intermediate	5.09E+11	28.1	5308.3	30.6	50.64	1.99E+11
	[5-15]	*Max Hydropower*	*6.04E+11*	*14.7*	*6387.3*	*16.5*	*27.45*	*1.04E+11*
		Max Ecological	4.34E+11	38.7	4446.9	41.9	68.61	2.74E+11
		Intermediate	5.28E+11	25.4	5344.4	30.1	49.58	1.80E+11

(1) Prices in Colombian Peso ($)

(2) Revenues and annual energy from Single Objective used as reference frame

Table 4-2. Multi-objective optimization results for the period 2010-2011

Optimization Type	**Outlet discharge range [m^3/s]**	**Solution**	**Revenues [$] (1)**	**Revenues reduction [%] (2)**	**Annual Energy [GWh]**	**Annual energy reduction [%] (2)**	**Average Daily Discharge in Guatapé River [m^3/s]**	**Ecological discharge value [$]**
[-]	[-]	Measured	8.25E+11	[-]	5721.3	[-]	12.02	[-]
Single Objective	[-]	Max Hydropower	9.40E+11	[-]	6480.8	[-]	0.00	[-]
Bi-Objective	[20-30]	Max Hydropower	7.45E+11	20.7	5166.1	20.3	26.45	1.94E+11
		Max Ecological	6.48E+11	31.1	4478.5	30.9	40.66	2.92E+11
		Intermediate	7.07E+11	24.7	4862.5	25.0	32.84	2.32E+11

(1) Prices in Colombian Peso ($)

(2) Revenues and annual energy from Single Objective used as reference frame

Table 4-3. Multi-objective optimization results for the period 2013-2014

Optimization Type	Outlet discharge range [m³/s]	Solution	Revenues [$] (1)	Revenues reduction [%] (2)	Annual Energy [GWh]	Annual energy reduction [%] (2)	Average Daily Discharge in Guatapé River [m³/s]	Ecological discharge value [$]
[-]	[-]	Measured	9.98E+11	[-]	5191.6	[-]	7.36	[-]
Single Objective	[-]	Max Hydropower	1.08E+12	[-]	5659.6	[-]	0.00	[-]
Bi-Objective	[10-30]	Max Hydropower	8.70E+11	19.7	4528.1	20.0	22.1	2.14E+11
		Max Ecological	7.48E+11	31.0	3974.5	29.8	33.14	3.36E+11
		Intermediate	8.13E+11	25.0	4184.3	26.1	29.05	2.71E+11

(1) Prices in Colombian Peso ($)

(2) Revenues and annual energy from Single Objective used as reference frame

Table 4-4. Multi-objective optimization results for the period 2015-2016

Scenario 3: Maximizing hydropower revenues, maximizing ecological discharge and minimizing flood risk via a Tri-objective optimization

A tri-objective optimization (maximizing hydropower revenues, maximizing ecological discharge and minimizing flood risk) was carried out for the simulation period 1. In this case, the optimization uses as input data the rating curve at La Pesca site (Figure 2-8) and the time-series of discharges at Samaná Norte river mouth, excluding the contributions from San Carlos Plant (Villada, 2021).

Given the available information, the Pareto optimal solutions (Figure 4-7) suggests that there is no risk of flood at La Pesca village in the period 2010-2011, i.e., the outcome is the same as the bi-objective case. To assess better this result, the discharges from the single-objective optimization were used to estimate the water levels at La Pesca (Figure 4-8). Even with daily discharges of 227 m^3/s from San Carlos Power Plant, the discharges at Samaná Norte river mouth never reached 800 m^3/s, thus, the water levels at the site were always below the warning level.

Several reasons were considered to explain why the outcomes showed no flood possibility. As stated in Chapter 2, the hydraulic model used to build the rating curve at La Pesca site has some limitations, which result in no flood risk at La Pesca when average discharges in the Magdalena river are being applied. To fairly reproduce the “La Niña” effects in the Samaná Norte watershed, the hydraulic model must include Magdalena river´s daily discharges (information currently not available to this research).

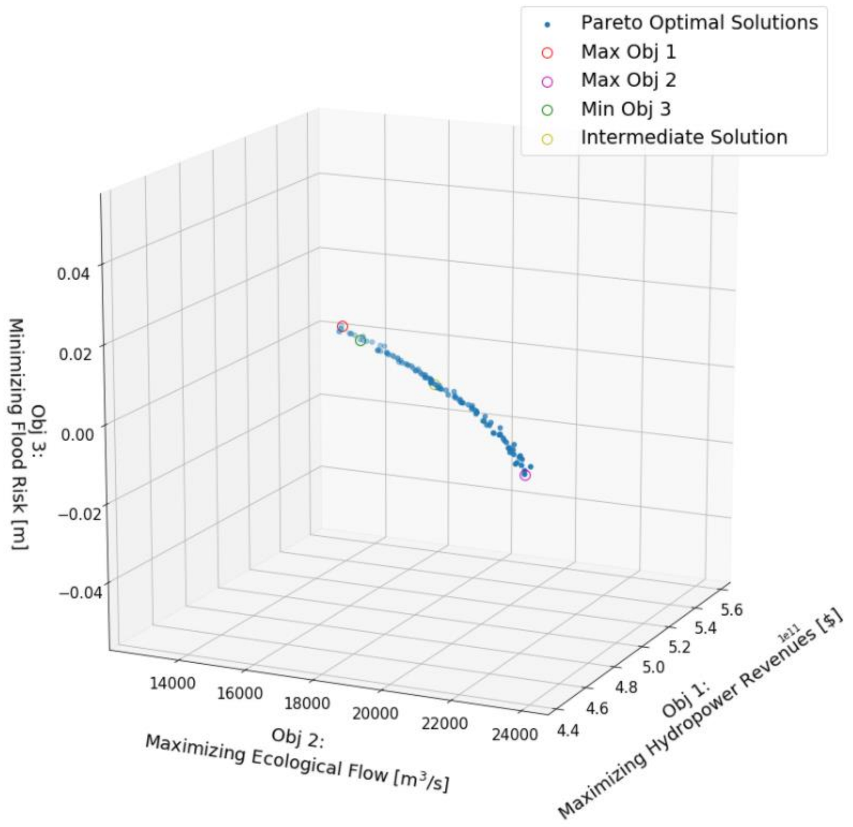


Figure 4-7. Pareto Optimal Solutions for tri-objective optimization, outlet discharge in the range 20-30 m^3/s, period 2010-2011

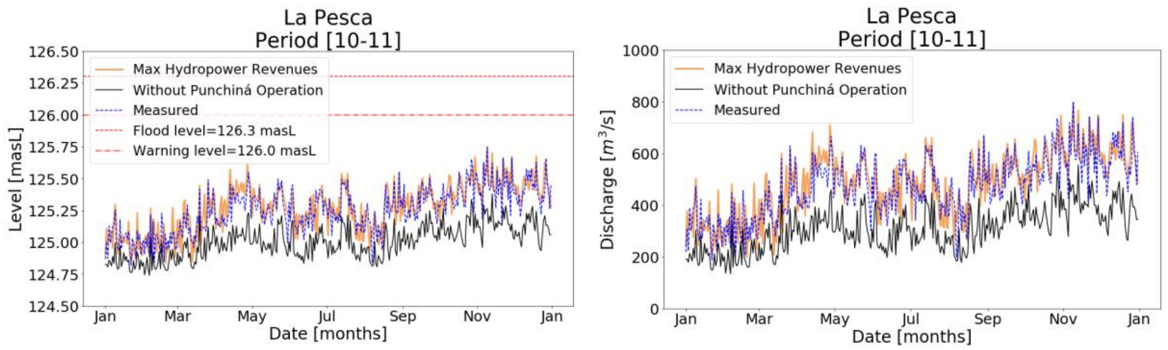


Figure 4-8. Daily behaviour at La Pesca site for single-objective optimization, period 2010-2011. Left: water levels. Right: Samaná Norte river discharge

5 Conclusions and recommendations

5.1 Conclusions

The case study for this research was the Punchiná reservoir and the San Carlos Hydroelectric Project in Colombia. The aim was to determine optimal operational rules for the Punchiná reservoir considering hydropower production, ecological discharge and flood mitigation, i.e., by allowing high revenues from hydroelectric generation while reducing the flood hazards downstream at La Pesca Village and allowing flow throughout the year in the Guatapé river reach, downstream of the dam by using Genetic Algorithm for the multi-objective optimization.

The results obtained from the combination of multi-objective optimization-simulation models made possible to answer the research question posed in Chapter 1:

What are the trade-offs between hydropower revenues, flood mitigation and ecological discharge in Punchiná reservoir?

In this section each sub-question is addressed individually.

What is the ecological discharge that can be taken from Punchiná reservoir without impacting the hydropower generation?

The bi-objective optimization showed that maximizing hydropower revenues and maximizing ecological discharges at Guatapé river, reach downstream of Punchiná dam, are competing objectives because the flow that goes directly into Guatapé river cannot be used for generating energy at San Carlos powerhouse. Therefore, it does not result in revenues for the generator company. If adding an ecological flow into Guatapé river is desired by the stakeholders, this will always come at a cost for the hydropower generation under the current scheme.

Chapter 4 results showed that although there is a reduction in revenues, there is also an ecosystem services gain. The improvement of these services are difficult to quantify but impacts positively the society and the environment. Even if the natural flows are low for a given year (for example during El Niño periods), there is enough resource to suffices both objectives by making trade-offs between the hydropower and the ecological flow.

Compromises can be conceived amongst the stakeholders to achieve both objectives. For instance, to make this approach attractive from the generator company point of view, the outlet discharge can be monetarized. The government- responsible for the water resources and watershed management- should develop an incentive system to pay for the ecological flow where the value of the ecosystemic services is the revenue lost in power generation. In this case, the objectives would not be competing since revenues will also be generated by the flow that does not go through the turbines.

Another solution is to build a small powerhouse just downstream of the dam, in the Guatapé stream. The outlet flow will actually generate energy all year long, thus, it reduces the impact on revenues and energy generation losses. The latter suggestion is a scheme applied worldwide to take advantage of the ecological flow discharge. A feasibility study must be conducted to assess the profitability of this solution. The result of this study may say if it is a good option for the generators or if the government should subsidize part of the refurbishment project.

Finally, both schemes would require the installation of a discharge monitoring device to measure the outlet flow.

How can the ecological discharge be provided in the Guatapé river, reach downstream of Punchiná dam?

Natural flow regimes exhibit time variability needed for the riverine ecosystems to develop properly by creating and maintaining the dynamics between the main channel and the floodplains (Poff et al., 1997). Therefore, the outflow discharge was defined as a range and not as a constant discharge to provide a variable flow along the Guatapé river reach and to enhance the ecological functions of the flow. Along all the report, the inclusion of an outlet discharge is mentioned. Currently there is no conduit that allows to discharge this flow. Therefore, it is necessary to build a structure that allows daily flow releases into Guatapé river. Since there is no possibility of adding a pipe into the dam´s body since the dam consists of an earthfill structure, the best approach seems to be a bypass tunnel. A detailed study – out of the scope of this research - must be carried out to assess the feasibility of constructing a tunnel and it must use as inputs the discharges defined in the present report to design the structure. Geotechnical and topographical studies of the site must be conducted as complement of the bypass-tunnel feasibility study.

Another option may be to construct a pumping station. Once again, a feasibility study must be conducted to assess the profitability of these solutions.

How should the San Carlos Power Plant be operated to mitigate the flood risk at La Pesca village?

Chapter 4 results revealed that the flood mitigation objective is not a competing objective against the hydropower and ecological flow objective when there are average flow conditions in the Magdalena river. Villada (2021) results from the Samaná Norte hydraulic model reveal that 2 situations must convey to have floods at La Pesca village; simultaneous high discharges at Samaná Norte and Magdalena river. Since the optimization was carried out with average flows in the Magdalena river, floods can always be avoided regardless of the releases from San Carlos Hydrolectric Plant. The main conclusion from this tri-objective optimization and from the current hydraulic model from Villada (2021) is that the Magdalena river creates a hydraulic control for the Samaná Norte river, affecting the water levels at La Pesca. Hence, to fairly reproduce the "La Niña" effects at La Pesca village, the hydraulic model must include Magdalena river´s daily discharges (information not available at the moment).

5.2 Recommendations

In this section, recommendations for further research of the Punchiná reservoir and the Samaná Norte river watershed are given based on the findings of the present study.

The main remark from the routing process of Punchiná reservoir was the evaluation of the available data in the Guatapé watershed. As explained thoroughly in Chapter 4, many problems arose during the routing in the period 2010-2017 due to the registered information. Furthermore, Villada (2021) while developing the hydraulic model of the Samaná Norte river in HEC-RAS, experienced also problems due to the lack of available daily data in the Samaná Norte watershed. This information déficit impacts largely the calibration of the hydraulic model and the outcomes, mainly during extreme events. To avoid these problems in the future and to produce more reliable results, the recommendation is to improve the monitoring system in the Samaná Norte river watershed including the hydroelectric developments located in this basin.

This research project focused on developing operation curves for the years 2010-2011, 2013-2014 and 2015-2016, taking these periods as representative for high, average and low flow conditions in the system. However, the inputs of a reservoir are of stochastic nature. Hence, an improvement of the model is to include the stochastic aspect of the external disturbance by developing synthetic time-series based on past measurements. This was out of the scope of the project.

As stated on the previous paragraph, the inputs of a reservoir are stochastic. Therefore, it is difficult to predict whether a year will present high, average or low flow conditions. Based on the bi-objective optimization results for the 3 simulation periods, where the outlet discharge is constraint according to Equation (24) and where the hydropower revenue generation was the prioritized objective, bounds for a general operational guide curve are recommended and showed in Figure 5-1. This solution was selected since the impact on revenues is 20% for the 3 meteorological conditions.

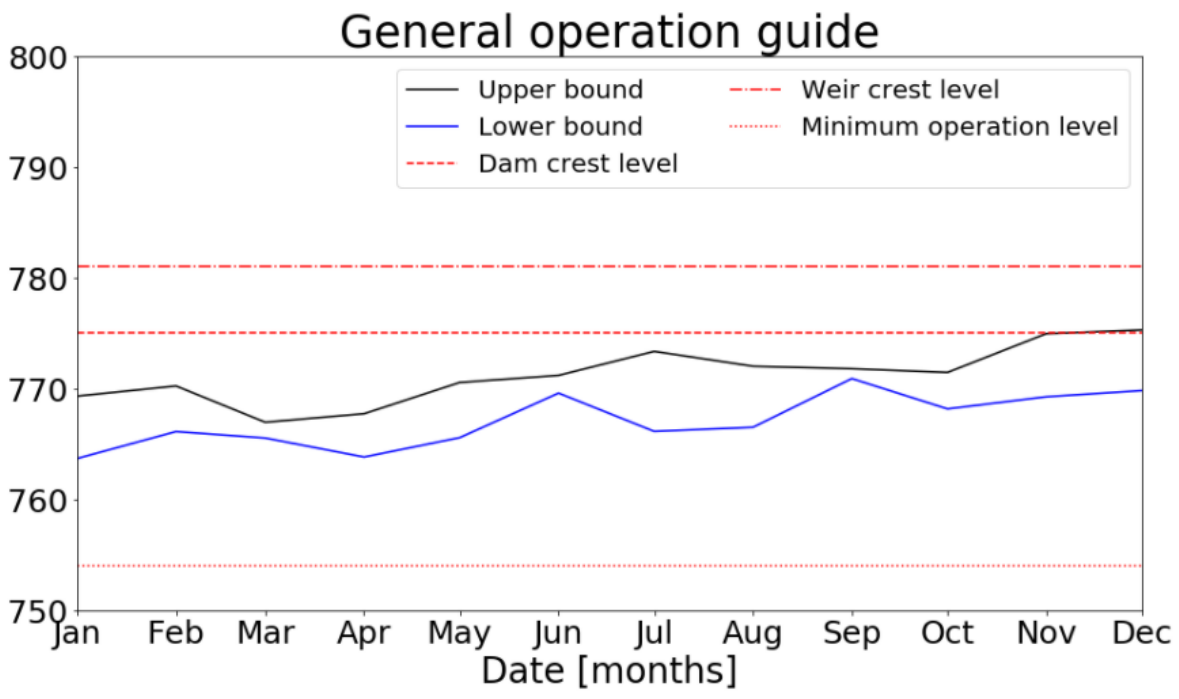


Figure 5-1. Upper and lower bound for a general operation guide

The multi-objective optimization allow to estimate average yearly discharges in Guatapé river, reach downstream of Punchiná dam by the inclusion of an outlet discharge and by letting more spill flow during the year. However, currently there is no study that details the necessary flow that must be present at Guatapé river. Therefore, a better estimation of the required ecological discharge must be carried out. This detail study will be conducted by Villada in his PhD dissertation intitled *Implementation of environmental flow regime for multicomponent hydropower generation*.

6 Bibliography

Acolgen—Colombian Association of Electric Power Generators,. (2019). https://www.acolgen.org.co/

Ahmad, A., El-Shafie, A., Razali, S. F. M., & Mohamad, Z. S. (2014). Reservoir Optimization in Water Resources: A Review. *Water Resources Management*, *28*(11), 3391-3405. https://doi.org/10.1007/s11269-014-0700-5

Banco de la República de Colombia. (s. f.). *Tasas de colocación por modalidad de crédito. 1.2.6. Histórico para un tipo de cuenta_periodicidad mensual*. Recuperado 13 de diciembre de 2020, de www.superfinanciera.gov.co/

Batimetría S.A.S. (2019). *Informe de levantamiento batimétrico del embalse de Punchiná* (p. 27).

Blank, J. (2020). *pymoo: Multi-objective Optimization in Python*. https://pymoo.org/index.html

Blank, J., & Deb, K. (2020). *Pymoo: Multi-Objective Optimization in Python*. *8*, 13.

Chow, V. T., Maidment, D. R., & Mays, L. W. (1988). *Applied Hydrology*. McGraw-Hill.

Consejo Nacional de Operación. (2018). *Acuerdo 1084.Acuerdo por el que se aprueba la incorporación de un cambio en el factor de conversión de la central hidroeléctrica San Carlos*.

Consejo Nacional de Operación. (2019). *Acuerdo 1200. Acuerdo por el cual se aprueba la incorporación de un cambio en el factor de conversión de las plantas de generación Porce II y Playas*.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182-197. https://doi.org/10.1109/4235.996017

Deb, Kalyanmoy. (2011). *Multi-Objective Optimization Using Evolutionary Algorithms: An Introduction*. 25.

Díaz Serna, F. J. (2011). *Optimización de la operación y evaluación de la eficiencia técnica de una empresa de generación hidroeléctrica en mercados de corto plazo* [Doctoral Dissertation]. Universidad Nacional de Colombia.

European Commission. (2014). *Ecological flows in the implementation of the Water Framework Directive. Draft final version*.

Fenómeno Niño y Niña—IDEAM. (s. f.). Recuperado 17 de enero de 2021, de http://www.siac.gov.co/web/siac/ninoynina

Fenton, J. D. (1992). Reservoir routing. *Hydrological Sciences Journal*, *37*(3), 233-246. https://doi.org/10.1080/02626669209492584

Integral -Proe S.A.S. (2013). *Integral (2013) Environmental Impact Assessment—APROVECHAMIENTO HIDROELÉCTRICO DEL RÍO SAMANÁ NORTE PROYECTO PORVENIR II, Celsia S.A, Medellín.*

ISAGEN. (2020, diciembre). https://www.isagen.com.co/en/home

Kora, P., & Yadlapalli, P. (2017). Crossover Operators in Genetic Algorithms: A Review. *International Journal of Computer Applications*, *162*(10), 34-36. https://doi.org/10.5120/ijca2017913370

Lin, Q. (2011). Influence of Dams on River Ecosystem and Its Countermeasures. *Journal of Water Resource and Protection*, *03*(01), 60-66. https://doi.org/10.4236/jwarp.2011.31007

Lugt, D. (2018). *Operating Sittaung's Reservoirs: A two-stage model predictive control method for managing a multi-reservoir system for hydropower, irrigation and flood mitigation* [Delft University of Technology]. http://resolver.tudelft.nl/uuid:8cc79e24-3bd7-4a40-9da8-98e232009a49

Malekmohammadi, B., Zahraie, B., & Kerachian, R. (2011). Ranking solutions of multi-objective reservoir operation optimization models using multi-criteria decision analysis. *Expert Systems with Applications*, *38*(6), 7851-7863. https://doi.org/10.1016/j.eswa.2010.12.119

Myo Lin, N., Tian, X., Rutten, M., Abraham, E., Maestre, J. M., & van de Giesen, N. (2020). Multi-Objective Model Predictive Control for Real-Time Operation of a Multi-Reservoir System. *Water*, *12*(7), 1898. https://doi.org/10.3390/w12071898

Poff, N. L., Allan, J. D., Bain, M. B., Karr, J. R., Prestegaard, K. L., Richter, B. D., Sparks, R. E., & Stromberg, J. C. (1997). The Natural Flow Regime. *BioScience*, *47*(11), 769-784. https://doi.org/10.2307/1313099

Reddy, M. J., & Kumar, D. N. (2006). Optimal Reservoir Operation Using Multi-Objective Evolutionary Algorithm. *Water Resources Management*, *20*(6), 861-878. https://doi.org/10.1007/s11269-005-9011-1

Tenenbaum, M., & Pollard, H. (1963). *Ordinary Differential Equations*. Dover Publications.

Villada, J. (2021). *Implementation of environmental flow regime for multicomponent hydropower generation (Unpublished)* [PhD Dissertation]. IHE Delft Institute for Water Education.

XM. (s. f.). *Portal*. http://portalbissrs.xm.com.co/

7 Appendix 1: Case Study Data

7.1 Conversion factors Punchiná and Las Playas reservoir

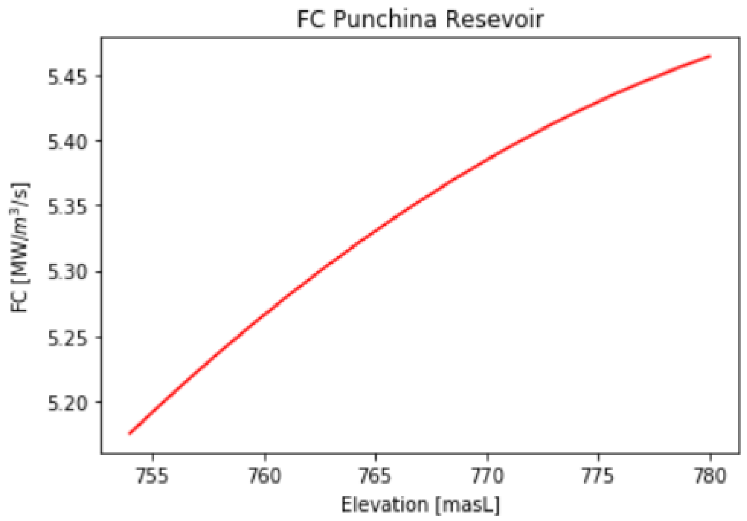


Figure 7-1. Conversion Factor curve in San Carlos Hydroelectric Project
Source: Consejo Nacional de Operación, (2018)

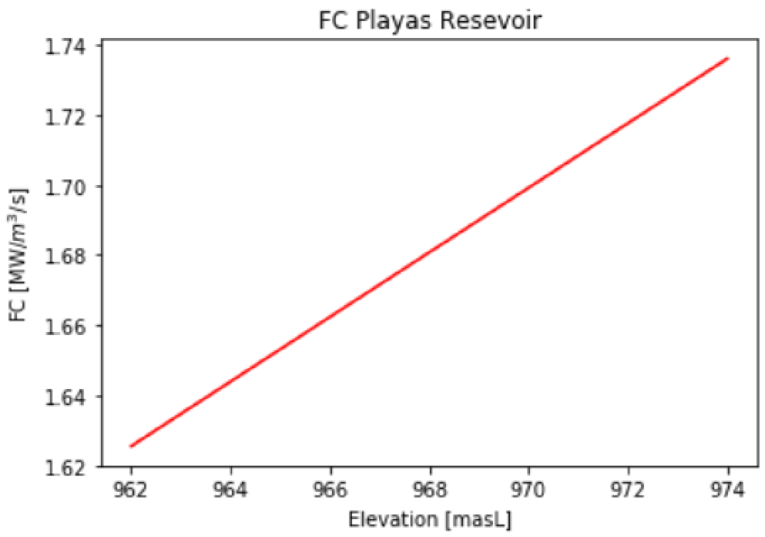


Figure 7-2. Conversion Factor curve in Playas Hydroelectric Project
Source: Consejo Nacional de Operación, (2019)

7.2 Energy prices and monthly nominal rates – period 2010-2017

Year/ Month	1	2	3	4	5	6	7	8	9	10	11	12
2010	152.3	195.6	190.0	195.5	150.3	91.3	82.9	84.9	112.3	136.5	92.0	66.7
2011	89.5	105.7	79.7	73.4	58.5	60.1	55.5	89.0	92.5	73.6	70.5	55.8
2012	53.6	78.5	119.1	57.0	46.9	87.4	79.2	140.3	183.4	198.0	167.0	180.3
2013	184.5	181.2	137.5	229.8	139.8	143.1	232.1	150.6	144.2	211.9	210.4	164.0
2014	160.5	188.0	151.3	373.7	382.0	334.9	184.7	200.2	177.3	207.0	166.6	175.1
2015	187.6	166.2	205.5	161.3	259.2	186.4	206.2	183.7	458.8	1106.6	767.0	630.2
2016	554.2	646.0	830.0	332.5	140.3	168.0	129.2	198.0	150.4	184.9	166.7	107.5
2017	111.5	157.0	119.9	96.5	71.7	68.5	70.6	92.6	124.3	137.4	126.7	101.3

Table 7-1. Monthly electricity prices in the period 2010-2017 in $ (Colombian peso)
Source: XM, (s. f.)

Year/ Month	1	2	3	4	5	6	7	8	9	10	11	12
2010	0.652	0.725	0.673	0.654	0.635	0.626	0.696	0.631	0.652	0.631	0.580	0.592
2011	0.788	0.669	0.637	0.749	0.735	0.794	0.843	0.824	0.826	0.811	0.822	0.815
2012	0.920	0.795	0.877	0.852	0.909	0.819	0.844	0.868	0.817	0.800	0.788	0.799
2013	0.795	0.744	0.754	0.669	0.596	0.651	0.676	0.602	0.709	0.668	0.666	0.652
2014	0.696	0.674	0.652	0.666	0.702	0.667	0.671	0.637	0.685	0.677	0.647	0.736
2015	0.756	0.669	0.682	0.687	0.707	0.694	0.717	0.695	0.697	0.658	0.822	0.820
2016	0.803	0.998	0.874	1.009	0.943	0.978	1.005	0.936	1.105	1.016	1.030	0.961
2017	1.046	1.006	0.936	0.865	0.843	0.849	0.824	0.766	0.790	0.755	0.723	0.728

Table 7-2. Monthly nominal rates (placement rates for corporative commercial credits for periods longer than 5 years) in the years 2010-2017, in Colombia in percentage
Source: Banco de la República de Colombia, (s. f.)

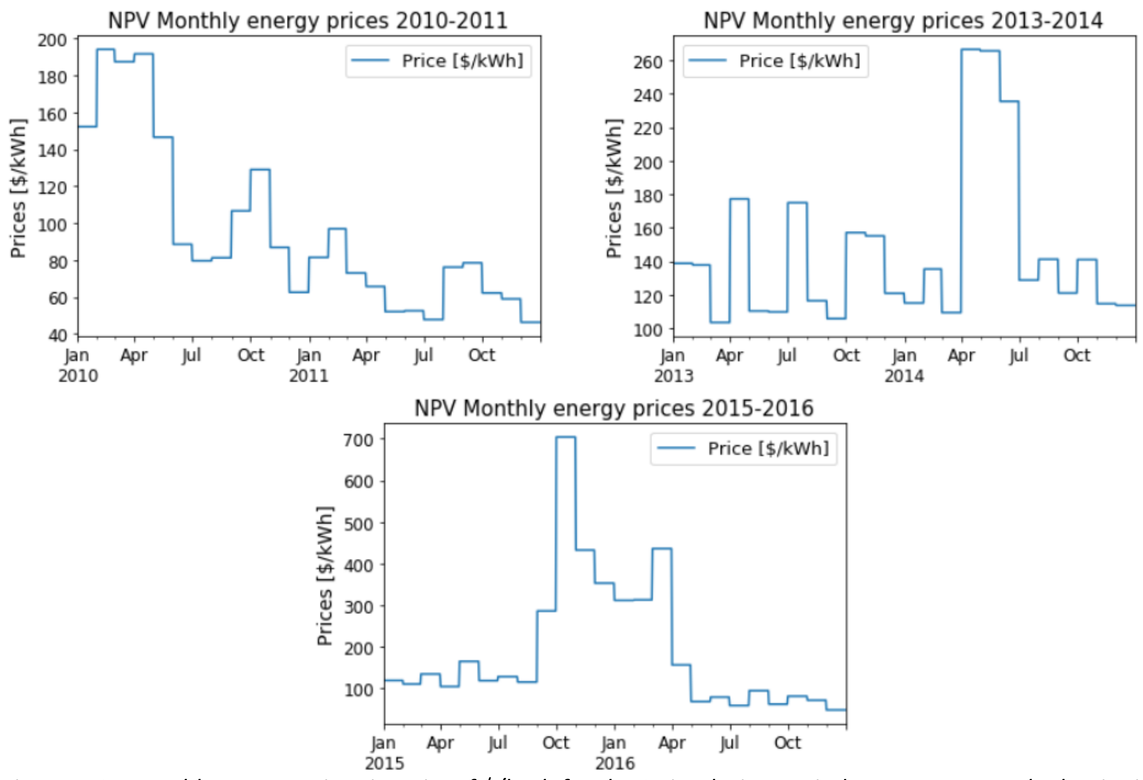


Figure 7-3. Monthly energy prices in units of $/kWh for the 3 simulation periods, set at NPV at the beginning of each simulation period ($ represents Colombian peso)

8 Appendix 2: Reservoir Routing Code

Routing

Function:

$$\frac{dS}{dt} = I(t) - Q(t, S)$$
$$\frac{dh}{dt} = \frac{I(t) - Q(t, h)}{A(h)}$$

Packages

```
import numpy as np
import pandas as pd
import math
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
%matplotlib inline
```

Folder Location

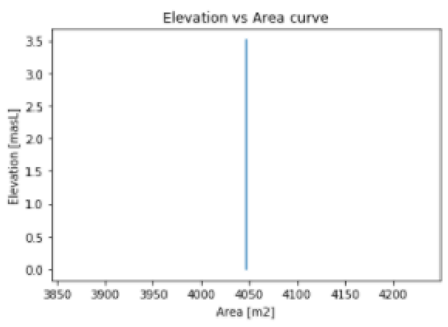
```
# Set folder
data_loc='D:\\users\\mairm\\Documents\\TU DELFT\\Additional Thesis\\Python\\Example\\'
```

Input Data

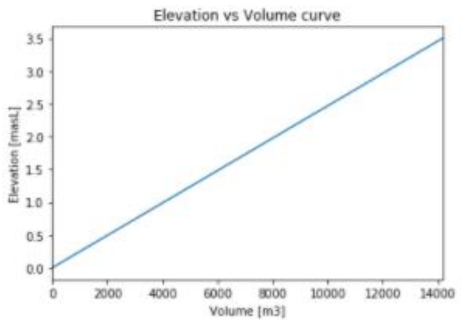
Reservoir curve

```
# Reservoir Curve -- Excel File

# 1. Upload data from Excel file into dataframe and convert it to numpy array
# In this case, the Excel file contains a Sheet with the following information per column:
# -1. Elevation [masL]
# -2. Area [m2]
# -3. Volume [m3]

data_res=pd.read_excel(data_loc+'Example.xlsx',sheet_name='Reservoir Curve')
np_data_res=data_res.to_numpy()
titled_res=list(data_res.columns.values)


# 2. Plot Data
# Elevation vs Area
plt.figure()
data_res.plot(x=titled_res[1],y=titled_res[0], legend=None)
plt.ylabel(titled_res[0])
plt.title('Elevation vs Area curve')

# Save Figure
plt.savefig(data_loc+'A-El.jpg')

# Volume vs Area
plt.figure()
data_res.plot(x=titled_res[2],y=titled_res[0], legend=None)
plt.ylabel(titled_res[0])
plt.title('Elevation vs Volume curve')

# Save Figure
plt.savefig(data_loc+'Vol-El.jpg')
```

```
<Figure size 432x288 with 0 Axes>
```

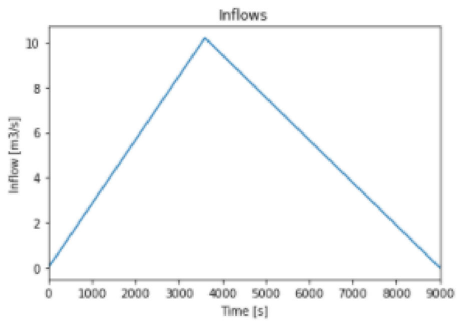


```
<Figure size 432x288 with 0 Axes>
```



Inflows

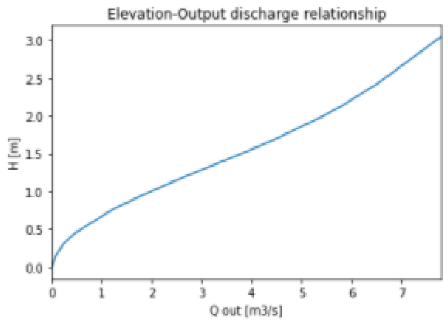
```
# Inflows

# 1. Upload data from Excel file into dataframe and convert it to numpy array
# In this case, the Excel file contains 1 Sheet with the following information per column:
# -1. Time [seconds]
# -2. Inflows [m3/s]
data_in=pd.read_excel(data_loc+'Example.xlsx',sheet_name='Inflows')
np_datain=data_in.to_numpy()
titled_in=list(data_in.columns.values)

# 2. Separate input data in different numpy arrays
in_time=np_datain[0:,0].astype(float)
in_In1=np_datain[0:,1].astype(float)

# 3. Plot Data
plt.figure()
data_in.plot(x=titled_in[0],y=titled_in[1], legend=None)
plt.ylabel(titled_in[1])
plt.title('Inflows')

# 4. Save figure
plt.savefig(data_loc+'Inflows.jpg')
```



Outflows

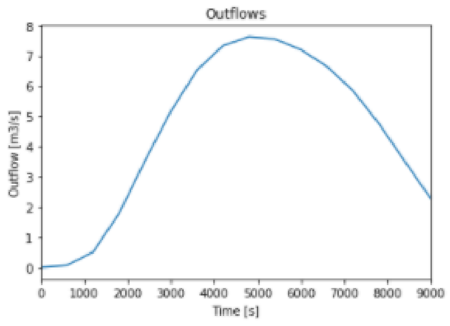
Elevation-Discharge relationship

```
# Outflows

# 1. Upload data from Excel file into dataframe and convert it to numpy array
# In this case, the Excel file contains 1 Sheet with the following information per column:
# -1. Elevation [masL]
# -2. Outflow [m3/s]
data_pout=pd.read_excel(data_loc+'Example.xlsx',sheet_name='Outflow operation')
np_datapout=data_pout.to_numpy()
titled_pout=list(data_pout.columns.values)

# 2. Plot Data
plt.figure()
data_pout.plot(x=titled_pout[1],y=titled_pout[0], legend=None)
plt.ylabel(titled_pout[0])
plt.title('Elevation-Output discharge relationship')

# 3. Save figure
plt.savefig(data_loc+'Outflows_paired_data.jpg')
```



Outflows time series

```
# Outflows

# 1. Upload data from Excel file into dataframe and convert it to numpy array
# In this case, the Excel file contains 1 Sheet with the following information per column:
# -1. Time [seconds]
# -2. Outflows [m3/s]
data_out=pd.read_excel(data_loc+'Example.xlsx',sheet_name='Outflows')
np_dataout=data_out.to_numpy()
titled_out=list(data_out.columns.values)

# 2. Separate output data in different numpy arrays
out_time=np_dataout[0:,0].astype(float)
out_Out1=np_dataout[0:,1].astype(float)

# 3. Plot Data
plt.figure()
data_out.plot(x=titled_out[0],y=titled_out[1], legend=None)
plt.ylabel(titled_out[1])
plt.title('Outflows')

# 4. Save figure
plt.savefig(data_loc+'Outflows.jpg')
```



Initial Values

```
# Initial water elevation
H0=0

# Time step in seconds
dt=600

# Number of Time steps
Tend=16
```

Outflow data type: Elevation-Discharge relationship

Numerical Methods

1. Euler

$$h_{n+1} = h_n + \Delta \frac{I(t_n) - Q(t_n, h_n)}{A(h_n)}$$

```
def EulerMethod1 (dt,Tend,H0,AERes,Qin1,Qout1):

#   Input Parameters:
#   dt=time step for computations [s]
#   Tend=number of simulation steps [-]
#   H0=initial reservoir elevation [masL]
#   AEres=matrix of 2 columns - Elevation [masL] and Surface Area [m^2]
#   Qin1=vector - inflow [m^3/s]
#   Qout1=matrix of 2 columns - Elevation [masL] and Discharge [m^3/s]

#   Output Parameters:
#   y=vector - reservoir levels [masL]
#   t=vector - time [s]
#   A=vector - surface elevation [m^2]
#   qin1=vector - inflow [m^3/s]
#   qout1=vector - outflow [m^3/s]

#   Initial Values
    y=np.zeros(Tend)
    y[0]=H0
    t=np.arange(0,Tend*dt,dt)
    qin1=Qin1
    qout1=np.zeros(Tend)
    qout1[0]=np.interp(H0,Qout1[0:,0],Qout1[0:,1])
    A=np.zeros(Tend)
    A[0]=np.interp(H0,AERes[0:,0],AERes[0:,1])

# Euler Method
    for n in range(0,Tend-1):
        y[n+1]=y[n]+dt*((qin1[n])-(qout1[n]))/A[n]
        qout1[n+1]=np.interp(y[n+1],Qout1[0:,0],Qout1[0:,1])
        A[n+1]=np.interp(y[n+1],AERes[0:,0],AERes[0:,1])

    return (y,t,A,qin1,qout1)
```

2. Modified Euler-Runge Kutta 2nd order-Predictor/Corrector Method

$$h_{n+1} = h_n + \frac{1}{2}(K_1 + K_2)$$

$$K_1 = \Delta \frac{I(t_n) - Q(t_n, h_n)}{A(h_n)}$$

$$K_2 = \Delta \frac{I(t_{n+1}) - Q(t_{n+1}, h_n + K_1)}{A(h_n + K_1)}$$

```
def EulerModMethod1 (dt,Tend,H0,AERes,Qin1,Qout1):

#   Input Parameters:
#   dt=Time step for computations [s]
#   Tend=number of simulation steps [-]
#   H0=initial reservoir elevation [masL]
#   AEres=matrix of 2 columns - Elevation [masL] and Surface Area [m^2]
#   Qin1=inflow [m^3/s]
#   Qout1=matrix of 2 columns - Elevation [masL] and Discharge [m^3/s]

#   Output Parameters:
#   y=vector - reservoir levels [masL]
#   t=vector - time [s]
#   A=vector - surface elevation [m^2]
#   qin1=vector - inflow [m^3/s]
#   qout1=vector - outflow [m^3/s]

#   Initial Values
    y=np.zeros(Tend)
    y[0]=H0
    t=np.arange(0,Tend*dt,dt)
    qin1=Qin1
    qout1=np.zeros(Tend)
    qout1[0]=np.interp(H0,Qout1[0:,0],Qout1[0:,1])
    A=np.zeros(Tend)
    A[0]=np.interp(H0,AERes[0:,0],AERes[0:,1])

# Modified Euler Method
    for n in range(0,Tend-1):
        k1=dt*((qin1[n])-(qout1[n]))/A[n]
        hn_k1=y[n]+k1
        An_hn_k1=np.interp(hn_k1,AERes[0:,0],AERes[0:,1])
        Q_hn_k1=np.interp(hn_k1,Qout1[0:,0],Qout1[0:,1])
        k2=dt*((qin1[n+1])-Q_hn_k1)/An_hn_k1
        y[n+1]=y[n]+0.5*(k1+k2)
        qout1[n+1]=np.interp(y[n+1],Qout1[0:,0],Qout1[0:,1])
        A[n+1]=np.interp(y[n+1],AERes[0:,0],AERes[0:,1])

    return (y,t,A,qin1,qout1)
```

3. Runge Kutta 3rd order

$$h_{n+1} = h_n + \frac{1}{6}(K_1 + 4K_2 + K_3)$$
$$K_1 = \Delta \frac{I(t_n) - Q(t_n, h_n)}{A(h_n)}$$
$$K_2 = \Delta \frac{I(t_{n+1/2}) - Q(t_{n+1/2}, h_n + 0.5K_1)}{A(h_n + 0.5K_1)}$$
$$K_3 = \Delta \frac{I(t_{n+1}) - Q(t_{n+1}, h_n - K_1 + 2K_2)}{A(h_n - K_1 + 2K_2)}$$

```python
def RK3Method1 (dt,Tend,H0,AERes,Qin1,Qout1):

#   Input Parameters:
#   dt=time step for computations [s]
#   Tend=number of simulation steps [-]
#   H0=initial reservoir elevation [masL]
#   AEres=matrix of 2 columns - Elevation [masL] and Surface Area [m^2]
#   Qin1=matrix of 2 columns - Time [s] and Discharge [m^3/s]
#   Qout1=matrix of 2 columns - Elevation [masL] and Discharge [m^3/s]

#   Output Parameters:
#   y=vector - reservoir levels [masL]
#   t=vector - time [s]
#   A=vector - surface elevation [m^2]
#   qin1=vector - inflow [m^3/s]
#   qout1=vector - outflow [m^3/s]

#   Initial Values
    y=np.zeros(Tend)
    y[0]=H0
    t=np.arange(0,Tend*dt,dt)
    qin1=Qin1[0:,1]
    qout1=np.zeros(Tend)
    qout1[0]=np.interp(H0,Qout1[0:,0],Qout1[0:,1])
    A=np.zeros(Tend)
    A[0]=np.interp(H0,AERes[0:,0],AERes[0:,1])

# Runge Kutta 3rd Order Method
    for n in range(0,Tend-1):
        k1=dt*((qin1[n])-(qout1[n]))/A[n]
        hn_05k1=y[n]+0.5*k1
        An_hn_05k1=np.interp(hn_05k1,AERes[0:,0],AERes[0:,1])
        tn=(t[n]+(t[n+1]-t[n])*0.5)
        Q_out_k2=np.interp(hn_05k1,Qout1[0:,0],Qout1[0:,1])
        Q_in_k2=np.interp(tn,Qin1[0:,0],Qin1[0:,1])
        k2=dt*(Q_in_k2-Q_out_k2)/An_hn_05k1
        hn_k1_2k2=y[n]-k1+2*k2
        An_hn_k1_2k2=np.interp(hn_k1_2k2,AERes[0:,0],AERes[0:,1])
        Qout_k3=np.interp(hn_k1_2k2,Qout1[0:,0],Qout1[0:,1])
        k3=dt*((qin1[n+1])-Qout_k3)/An_hn_k1_2k2
        y[n+1]=y[n]+(k1+4*k2+k3)/6
        qout1[n+1]=np.interp(y[n+1],Qout1[0:,0],Qout1[0:,1])
        A[n+1]=np.interp(y[n+1],AERes[0:,0],AERes[0:,1])

    return (y,t,A,qin1,qout1)
```

4. Runge Kutta 4th order

$$h_{n+1} = h_n + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4)$$
$$K_1 = \Delta \frac{I(t_n) - Q(t_n, h_n)}{A(h_n)}$$
$$K_2 = \Delta \frac{I(t_{n+1/2}) - Q(t_{n+1/2}, h_n + 0.5K_1)}{A(h_n + 0.5K_1)}$$
$$K_3 = \Delta \frac{I(t_{n+1/2}) - Q(t_{n+1/2}, h_n + 0.5K_2)}{A(h_n + 0.5K_2)}$$
$$K_4 = \Delta \frac{I(t_{n+1}) - Q(t_{n+1}, h_n + K_3)}{A(h_n + K_3)}$$

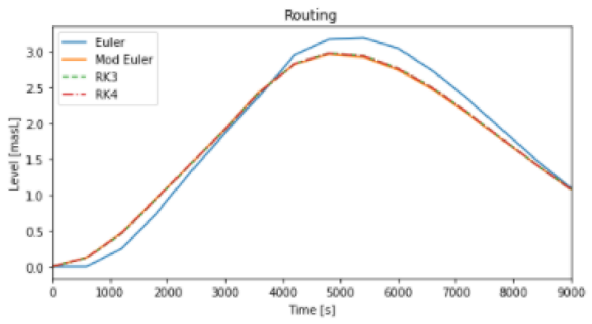
```
def RK4Method1 (dt,Tend,H0,AERes,Qin1,Qout1):

#   Input Parameters:
#   dt=Time step for computations [s]
#   Tend=number of simulation steps [-]
#   H0=initial reservoir elevation [masl]
#   AEres=matrix of 2 columns - Elevation [masL] and Surface Area [m^2]
#   Qin1=matrix of 2 columns - Time [s] and Discharge [m^3/s]
#   Qout1=matrix of 2 columns - Elevation [masL] and Discharge [m^3/s]

#   Output Parameters:
#   y=vector - reservoir levels [masL]
#   t=vector - time [s]
#   A=vector - surface elevation [m^2]
#   qin1=vector - inflow [m^3/s]
#   qout1=vector - outflow [m^3/s]

#   Initial Values
    y=np.zeros(Tend)
    y[0]=H0
    t=np.arange(0,Tend*dt,dt)
    qin1=Qin1[0:,1]
    qout1=np.zeros(Tend)
    qout1[0]=np.interp(H0,Qout1[0:,0],Qout1[0:,1])
    A=np.zeros(Tend)
    A[0]=np.interp(H0,AERes[0:,0],AERes[0:,1])

# Runge Kutta 4th Order Method
    for n in range(0,Tend-1):
        k1=dt*((qin1[n])-(qout1[n]))/A[n]
        hn_05k1=y[n]+0.5*k1
        An_hn_05k1=np.interp(hn_05k1,AERes[0:,0],AERes[0:,1])
        tn=(t[n]+(t[n+1]-t[n])*0.5)
        Q_out_k2=np.interp(hn_05k1,Qout1[0:,0],Qout1[0:,1])
        Q_in_k2=np.interp(tn,Qin1[0:,0],Qin1[0:,1])
        k2=dt*(Q_in_k2-Q_out_k2)/An_hn_05k1
        hn_05k2=y[n]+0.5*k2
        An_hn_05k2=np.interp(hn_05k2,AERes[0:,0],AERes[0:,1])
        Q_out_k3=np.interp(hn_05k2,Qout1[0:,0],Qout1[0:,1])
        k3=dt*(Q_in_k2-Q_out_k3)/An_hn_05k2
        hn_k3=y[n]+k3
        An_hn_k3=np.interp(hn_k3,AERes[0:,0],AERes[0:,1])
        Q_out_k4=np.interp(hn_k3,Qout1[0:,0],Qout1[0:,1])
        k4=dt*((qin1[n+1])-Q_out_k4)/An_hn_k3
        y[n+1]=y[n]+(k1+2*k2+2*k3+k4)/6
        qout1[n+1]=np.interp(y[n+1],Qout1[0:,0],Qout1[0:,1])
        A[n+1]=np.interp(y[n+1],AERes[0:,0],AERes[0:,1])

    return (y,t,A,qin1,qout1)
```

Routing

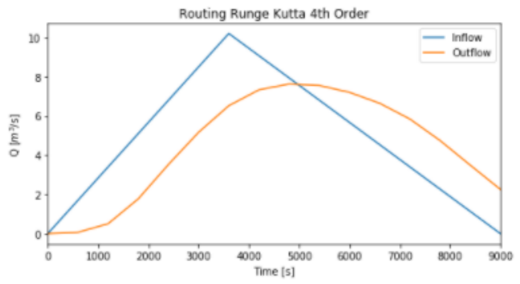
```
# 1. Run all numerical methods
y_e,t_e,A_e,qin1_e,qout1_e=EulerMethod1 (dt,Tend,H0,np_data_res,in_In1,np_datapout)
y_em,t_em,A_em,qin1_em,qout1_em=EulerModMethod1 (dt,Tend,H0,np_data_res,in_In1,np_datapout)
y_rk3,t_rk3,A_rk3,qin1_rk3,qout1_rk3=RK3Method1 (dt,Tend,H0,np_data_res,np_datain,np_datapout)
y_rk4,t_rk4,A_rk4,qin1_rk4,qout1_rk4=RK4Method1 (dt,Tend,H0,np_data_res,np_datain,np_datapout)

# 2.  Store results in Dataframes
Results_Euler=pd.DataFrame({'Time [s]':t_e,'Reservoir Level [m]':y_e,'Q_in [$m^3$/s]':qin1_e,'Q_out [$m^3$/s]':qout1_e})
Results_Mod_Euler=pd.DataFrame({'Time [s]':t_em,'Reservoir Level [m]':y_em,'Q_in [$m^3$/s]':qin1_em,'Q_out [$m^3$/s]':qout1_em})
Results_RK3=pd.DataFrame({'Time [s]':t_rk3,'Reservoir Level [m]':y_rk3,'Q_in [$m^3$/s]':qin1_rk3,'Q_out [$m^3$/s]':qout1_rk3})
Results_RK4=pd.DataFrame({'Time [s]':t_rk4,'Reservoir Level [m]':y_rk4,'Q_in [$m^3$/s]':qin1_rk4,'Q_out [$m^3$/s]':qout1_rk4})
Results_Summary=pd.DataFrame({'Time [s]':t_e,'Euler':y_e,'Modified Euler':y_em,'Runge Kutta 3rd order':y_rk3,'Runge Kutta 4th or
der':y_rk4})

# 3. Save all results to Excel
with pd.ExcelWriter('Output_Example_PD.xlsx') as writer:
    Results_Euler.to_excel(writer,sheet_name='Euler')
    Results_Mod_Euler.to_excel(writer,sheet_name='Modified Euler')
    Results_RK3.to_excel(writer,sheet_name='RK3')
    Results_RK4.to_excel(writer,sheet_name='RK4')
    Results_Summary.to_excel(writer,sheet_name='Summary')
```

Results Visualization

```
## Plot results from all numerical methods
f = plt.figure(figsize=(8, 4))
ax = f.add_subplot(111)
Results_Euler.plot(x='Time [s]',y='Reservoir Level [m]',ax=ax)
Results_Mod_Euler.plot(x='Time [s]',y='Reservoir Level [m]',ax=ax)
Results_RK3.plot(x='Time [s]',y='Reservoir Level [m]',linestyle='--',ax=ax)
Results_RK4.plot(x='Time [s]',y='Reservoir Level [m]',linestyle='-.',ax=ax)
ax.legend(['Euler','Mod Euler','RK3','RK4'])
plt.ylabel('Level [masL]')
plt.title('Routing')

# Save figure
plt.savefig(data_loc+'Results all mehtods - PD.jpg')
```



```
## Plot results from RK4 - Inflow and outflow hydrograph
f = plt.figure(figsize=(8, 4))
ax = f.add_subplot(111)
Results_RK4.plot(x='Time [s]',y='Q_in [$m^3$/s]',ax=ax)
Results_RK4.plot(x='Time [s]',y='Q_out [$m^3$/s]',ax=ax)
ax.legend(['Inflow','Outflow'])
plt.ylabel('Q [$m^3$/s]')
plt.title('Routing Runge Kutta 4th Order')

# Save figure
plt.savefig(data_loc+'Results RK4 - Inflow and outflow hydrogrphs - PD.jpg')
```

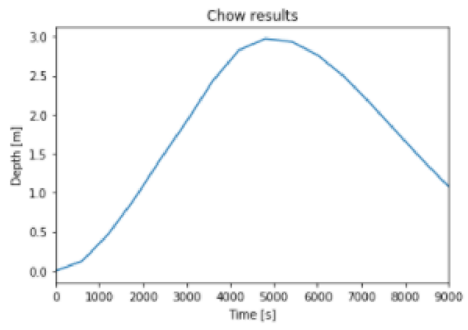


Results

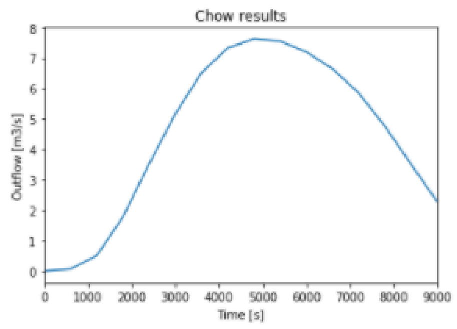
Chow et al, 1988

```
# 1. Upload data from Excel file into dataframe
# In this case, the Excel file contains 1 Sheet with the following information per column:
# -1. Time [seconds]
# -2. Depth [m]
# -2. Outflow [m3/s]
data_chow=pd.read_excel(data_loc+'Example.xlsx',sheet_name='Chow Results')
titled_chow=list(data_chow.columns.values)
np_datachow=data_chow.to_numpy()

# 2. Separate output data in different numpy arrays
chow_y=np_datachow[0:,1].astype(float)
chow_qout=np_datachow[0:,2].astype(float)

# 2. Plot Data
plt.figure()
data_chow.plot(x=titled_chow[0],y=titled_chow[1], legend=None)
plt.ylabel(titled_chow[1])
plt.title('Chow results')

plt.figure()
data_chow.plot(x=titled_chow[0],y=titled_chow[2], legend=None)
plt.ylabel(titled_chow[2])
plt.title('Chow results')
```

```
<Figure size 432x288 with 0 Axes>
```

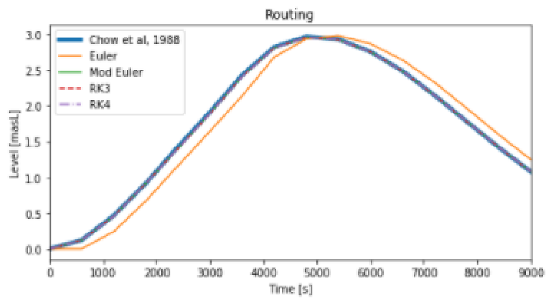


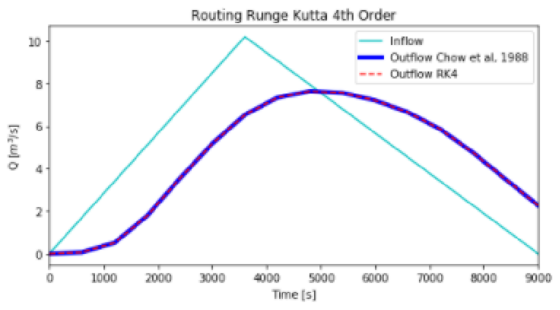
```
<Figure size 432x288 with 0 Axes>
```



Comparing Chow et al, 1988 results against Python routing

```
Results_all=pd.DataFrame({'Time [s]':t_e,'Chow et al, 1988':chow_y,'Euler':y_e,'Modified Euler':y_em,'Runge Kutta 3rd order':y_r
k3,'Runge Kutta 4th order':y_rk4})
Results_QoutRk4C=pd.DataFrame({'Time [s]':t_rk4,'Q_in [$m^3$/s]':qin1_rk4,'Q_out RK4 [$m^3$/s]':qout1_rk4, 'Q_out Chow [$m^3$/
s]':chow_qout})

# Results from all numerical methods and Chow et al, 1988
f = plt.figure(figsize=(8, 4))
ax = f.add_subplot(111)
Results_all.plot(x='Time [s]',y='Chow et al, 1988',linewidth=4,ax=ax)
Results_all.plot(x='Time [s]',y='Euler',ax=ax)
Results_all.plot(x='Time [s]',y='Modified Euler',ax=ax)
Results_all.plot(x='Time [s]',y='Runge Kutta 3rd order',linestyle='--',ax=ax)
Results_all.plot(x='Time [s]',y='Runge Kutta 4th order',linestyle='-.',ax=ax)
ax.legend(['Chow et al, 1988','Euler','Mod Euler','RK3','RK4'])
plt.ylabel('Level [masL]')
plt.title('Routing')

f = plt.figure(figsize=(8, 4))
ax = f.add_subplot(111)
Results_QoutRk4C.plot(x='Time [s]',y='Q_in [$m^3$/s]',color='c',ax=ax)
Results_QoutRk4C.plot(x='Time [s]',y='Q_out Chow [$m^3$/s]',color='b',linewidth=4,ax=ax)
Results_QoutRk4C.plot(x='Time [s]',y='Q_out RK4 [$m^3$/s]',color='r',linestyle='--',ax=ax)
ax.legend(['Inflow','Outflow Chow et al, 1988', 'Outflow RK4'])
plt.ylabel('Q [$m^3$/s]')
plt.title('Routing Runge Kutta 4th Order')
```





Outflow data type: Time series

Numerical Methods

1. Euler

$$h_{n+1} = h_n + \Delta \frac{I(t_n) - Q(t_n, h_n)}{A(h_n)}$$

```python
def EulerMethod2 (dt,Tend,H0,AERes,Qin1,Qout1):

#   Input Parameters:
#   dt=Time step for computations [s]
#   Tend=number of simulation steps [-]
#   H0=initial reservoir elevation [masL]
#   AEres=matrix of 2 columns - Elevation [masL] and Surface Area [m^2]
#   Qin1=vector - inflow [m^3/s]
#   Qout1= vector - outflow [m^3/s]

#   Output Parameters:
#   y=vector - reservoir levels [masL]
#   t=vector - time [s]
#   A=vector - surface elevation [m^2]
#   qin1=vector - inflow [m^3/s]
#   qout1=vector - outflow [m^3/s]

#   Initial Values
    y=np.zeros(Tend)
    y[0]=H0
    t=np.arange(0,Tend*dt,dt)
    qin1=Qin1
    qout1=Qout1
    A=np.zeros(Tend)
    A[0]=np.interp(H0,AERes[0:,0],AERes[0:,1])

# Euler Method
    for n in range(0,Tend-1):
        y[n+1]=y[n]+dt*((qin1[n])-(qout1[n]))/A[n]
        A[n+1]=np.interp(y[n+1],AERes[0:,0],AERes[0:,1])

    return (y,t,A,qin1,qout1)
```

2. Modified Euler-Runge Kutta 2nd order-Predictor/Corrector Method

$$h_{n+1} = h_n + \frac{1}{2}(K_1 + K_2)$$
$$K_1 = \Delta \frac{I(t_n) - Q(t_n, h_n)}{A(h_n)}$$
$$K_2 = \Delta \frac{I(t_{n+1}) - Q(t_{n+1}, h_n + K_1)}{A(h_n + K_1)}$$

```python
def EulerModMethod2 (dt,Tend,H0,AERes,Qin1,Qout1):

#   Input Parameters:
#   dt=Time step for computations [s]
#   Tend=number of simulation steps [-]
#   H0=initial reservoir elevation [masL]
#   AEres=matrix of 2 columns - Elevation [masL] and Surface Area [m^2]
#   Qin1= vector - inflow [m^3/s]
#   Qout1=vector - outflow [m^3/s]

#   Output Parameters:
#   y=vector - reservoir levels [masL]
#   t=vector - time [s]
#   A=vector - surface elevation [m^2]
#   qin1=vector - inflow [m^3/s]
#   qout1=vector - outflow [m^3/s]

#   Initial Values
    y=np.zeros(Tend)
    y[0]=H0
    t=np.arange(0,Tend*dt,dt)
    qin1=Qin1
    qout1=Qout1
    A=np.zeros(Tend)
    A[0]=np.interp(H0,AERes[0:,0],AERes[0:,1])

# Modified Euler Method
    for n in range(0,Tend-1):
        k1=dt*((qin1[n])-(qout1[n]))/A[n]
        hn_k1=y[n]+k1
        An_hn_k1=np.interp(hn_k1,AERes[0:,0],AERes[0:,1])
        k2=dt*((qin1[n+1])-qout1[n+1])/An_hn_k1
        y[n+1]=y[n]+0.5*(k1+k2)
        A[n+1]=np.interp(y[n+1],AERes[0:,0],AERes[0:,1])

    return (y,t,A,qin1,qout1)
```

3. Runge Kutta 4th order

$$h_{n+1} = h_n + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4)$$
$$K_1 = \Delta\frac{I(t_n) - Q(t_n, h_n)}{A(h_n)}$$
$$K_2 = \Delta\frac{I(t_{n+1/2}) - Q(t_{n+1/2}, h_n + 0.5K_1)}{A(h_n + 0.5K_1)}$$
$$K_3 = \Delta\frac{I(t_{n+1/2}) - Q(t_{n+1/2}, h_n + 0.5K_2)}{A(h_n + 0.5K_2)}$$
$$K_4 = \Delta\frac{I(t_{n+1}) - Q(t_{n+1}, h_n + K_3)}{A(h_n + K_3)}$$

```
def RK3Method2 (dt,Tend,H0,AERes,Qin1,Qout1):

#   Input Parameters:
#   dt=Time step for computations [s]
#   Tend=number of simulation steps [-]
#   H0=initial reservoir elevation [masl]
#   AEres=matrix of 2 columns - Elevation [masl] and Surface Area [m^2]
#   Qin1=matrix of 2 columns - Time [s] and Discharge [m^3/s]
#   Qout1=matrix of 2 columns - Time [s] and Discharge [m^3/s]

#   Output Parameters:
#   y=vector - reservoir levels [masl]
#   t=vector - time [s]
#   A=vector - surface elevation [m^2]
#   qin1=vector - inflow [m^3/s]
#   qout1=vector - outflow [m^3/s]

#   Initial Values
    y=np.zeros(Tend)
    y[0]=H0
    t=np.arange(0,Tend*dt,dt)
    qin1=Qin1[0:,1]
    qout1=Qout1[0:,1]
    A=np.zeros(Tend)
    A[0]=np.interp(H0,AERes[0:,0],AERes[0:,1])

# Runge Kutta 3rd Order Method
    for n in range(0,Tend-1):
        k1=dt*((qin1[n])-(qout1[n]))/A[n]
        hn_05k1=y[n]+0.5*k1
        An_hn_05k1=np.interp(hn_05k1,AERes[0:,0],AERes[0:,1])
        tn=(t[n]+(t[n+1]-t[n])*0.5)
        Q_out_k2=np.interp(tn,Qout1[0:,0],Qout1[0:,1])
        Q_in_k2=np.interp(tn,Qin1[0:,0],Qin1[0:,1])
        k2=dt*(Q_in_k2-Q_out_k2)/An_hn_05k1
        hn_k1_2k2=y[n]-k1+2*k2
        An_hn_k1_2k2=np.interp(hn_k1_2k2,AERes[0:,0],AERes[0:,1])
        k3=dt*((qin1[n+1])-(qout1[n+1]))/An_hn_k1_2k2
        y[n+1]=y[n]+(k1+4*k2+k3)/6
        A[n+1]=np.interp(y[n+1],AERes[0:,0],AERes[0:,1])

    return (y,t,A,qin1,qout1)
```

4. Runge Kutta 4th order

$$h_{n+1} = h_n + \frac{1}{6}(K_1 + 2K_2 + 2K_3 + K_4)$$
$$K_1 = \Delta\frac{I(t_n) - Q(t_n, h_n)}{A(h_n)}$$
$$K_2 = \Delta\frac{I(t_{n+1/2}) - Q(t_{n+1/2}, h_n + 0.5K_1)}{A(h_n + 0.5K_1)}$$
$$K_3 = \Delta\frac{I(t_{n+1/2}) - Q(t_{n+1/2}, h_n + 0.5K_2)}{A(h_n + 0.5K_2)}$$
$$K_4 = \Delta\frac{I(t_{n+1}) - Q(t_{n+1}, h_n + K_3)}{A(h_n + K_3)}$$

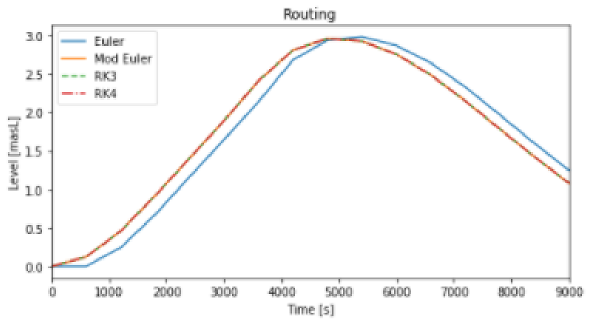
```python
def RK4Method2 (dt,Tend,H0,AERes,Qin1,Qout1):

#   Input Parameters:
#   dt=Time step for computations [s]
#   Tend=number of simulation steps [-]
#   H0=initial reservoir elevation [masL]
#   AEres=matrix of 2 columns - Elevation [masL] and Surface Area [m^2]
#   Qin1=matrix of 2 columns - Time [s] and Discharge [m^3/s]
#   Qout1=matrix of 2 columns - Time [s] and Discharge [m^3/s]

#   Initial Values
    y=np.zeros(Tend)
    y[0]=H0
    t=np.arange(0,Tend*dt,dt)
    qin1=Qin1[0:,1]
    qout1=Qout1[0:,1]
    A=np.zeros(Tend)
    A[0]=np.interp(H0,AERes[0:,0],AERes[0:,1])

# Runge Kutta 4th Order Method
    for n in range(0,Tend-1):
        k1=dt*((qin1[n])-(qout1[n]))/A[n]
        hn_05k1=y[n]+0.5*k1
        An_hn_05k1=np.interp(hn_05k1,AERes[0:,0],AERes[0:,1])
        tn=(t[n]+(t[n+1]-t[n])*0.5)
        Q_out_k2=np.interp(tn,Qout1[0:,0],Qout1[0:,1])
        Q_in_k2=np.interp(tn,Qin1[0:,0],Qin1[0:,1])
        k2=dt*(Q_in_k2-Q_out_k2)/An_hn_05k1
        hn_05k2=y[n]+0.5*k2
        An_hn_05k2=np.interp(hn_05k2,AERes[0:,0],AERes[0:,1])
        k3=dt*(Q_in_k2-Q_out_k2)/An_hn_05k2
        hn_k3=y[n]+k3
        An_hn_k3=np.interp(hn_k3,AERes[0:,0],AERes[0:,1])
        k4=dt*((qin1[n+1])-(qout1[n+1]))/An_hn_k3
        y[n+1]=y[n]+(k1+2*k2+2*k3+k4)/6
        A[n+1]=np.interp(y[n+1],AERes[0:,0],AERes[0:,1])

    return (y,t,A,qin1,qout1)
```

Routing

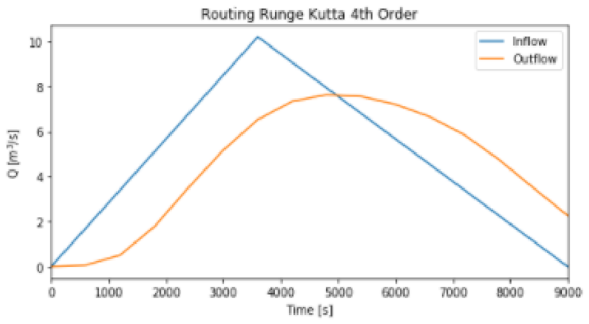
```python
# 1. Run all numerical methods
y_e,t_e,A_e,qin1_e,qout1_e=EulerMethod2 (dt,Tend,H0,np_data_res,in_In1,out_Out1)
y_em,t_em,A_em,qin1_em,qout1_em=EulerModMethod2 (dt,Tend,H0,np_data_res,in_In1,out_Out1)
y_rk3,t_rk3,A_rk3,qin1_rk3,qout1_rk3=RK3Method2 (dt,Tend,H0,np_data_res,np_datain,np_dataout)
y_rk4,t_rk4,A_rk4,qin1_rk4,qout1_rk4=RK4Method2 (dt,Tend,H0,np_data_res,np_datain,np_dataout)

# 2.  Store results in Dataframes
Results_Euler=pd.DataFrame({'Time [s]':t_e,'Reservoir Level [m]':y_e,'Q_in [$m^3$/s]':qin1_e,'Q_out [$m^3$/s]':qout1_e})
Results_Mod_Euler=pd.DataFrame({'Time [s]':t_em,'Reservoir Level [m]':y_em,'Q_in [$m^3$/s]':qin1_em,'Q_out [$m^3$/s]':qout1_em})
Results_RK3=pd.DataFrame({'Time [s]':t_rk3,'Reservoir Level [m]':y_rk3,'Q_in [$m^3$/s]':qin1_rk3,'Q_out [$m^3$/s]':qout1_rk3})
Results_RK4=pd.DataFrame({'Time [s]':t_rk4,'Reservoir Level [m]':y_rk4,'Q_in [$m^3$/s]':qin1_rk4,'Q_out [$m^3$/s]':qout1_rk4})
Results_Summary=pd.DataFrame({'Time [s]':t_e,'Euler':y_e,'Modified Euler':y_em,'Runge Kutta 3rd order':y_rk3,'Runge Kutta 4th or
der':y_rk4})

# 3. Save all results to Excel
with pd.ExcelWriter('Output_Example_TS.xlsx') as writer:
    Results_Euler.to_excel(writer,sheet_name='Euler')
    Results_Mod_Euler.to_excel(writer,sheet_name='Modified Euler')
    Results_RK3.to_excel(writer,sheet_name='RK3')
    Results_RK4.to_excel(writer,sheet_name='RK4')
    Results_Summary.to_excel(writer,sheet_name='Summary')
```

Results Visualization

```python
## Plot results from all numerical methods
f = plt.figure(figsize=(8, 4))
ax = f.add_subplot(111)
Results_Euler.plot(x='Time [s]',y='Reservoir Level [m]',ax=ax)
Results_Mod_Euler.plot(x='Time [s]',y='Reservoir Level [m]',ax=ax)
Results_RK3.plot(x='Time [s]',y='Reservoir Level [m]',linestyle='--',ax=ax)
Results_RK4.plot(x='Time [s]',y='Reservoir Level [m]',linestyle='-.',ax=ax)
ax.legend(['Euler','Mod Euler','RK3','RK4'])
plt.ylabel('Level [masL]')
plt.title('Routing')

# Save figure
plt.savefig(data_loc+'Results all mehtods - TS.jpg')
```



```
## Plot results from RK4 - Inflow and outflow hydrograph
f = plt.figure(figsize=(8, 4))
ax = f.add_subplot(111)
Results_RK4.plot(x='Time [s]',y='Q_in [$m^3$/s]',ax=ax)
Results_RK4.plot(x='Time [s]',y='Q_out [$m^3$/s]',ax=ax)
ax.legend(['Inflow','Outflow'])
plt.ylabel('Q [$m^3$/s]')2
plt.title('Routing Runge Kutta 4th Order')

# Save figure
plt.savefig(data_loc+'Results RK4 - Inflow and outflow hydrogrphs - TS.jpg')
```

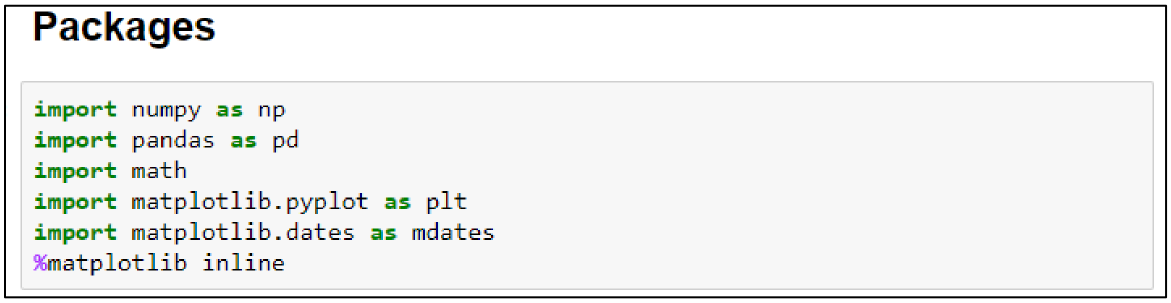


9 Appendix 3: Description of Python Script

A detailed description of how to use the program is given in this section followed by an example. The complete code with the example is attached in Appendix 2: Reservoir Routing Code.

The program contains the following:

1. Import libraries: numpy, pandas, math, matplotlib

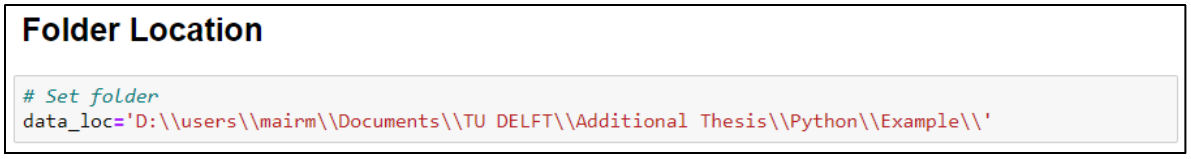


Packages

```
import numpy as np
import pandas as pd
import math
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
%matplotlib inline
```

Figure 9-1. Packages imported for Routing Script

2. Define folder location: where input files and output files are stored



Folder Location

```
# Set folder
data_loc='D:\\users\\mairm\\Documents\\TU DELFT\\Additional Thesis\\Python\\Example\\'
```

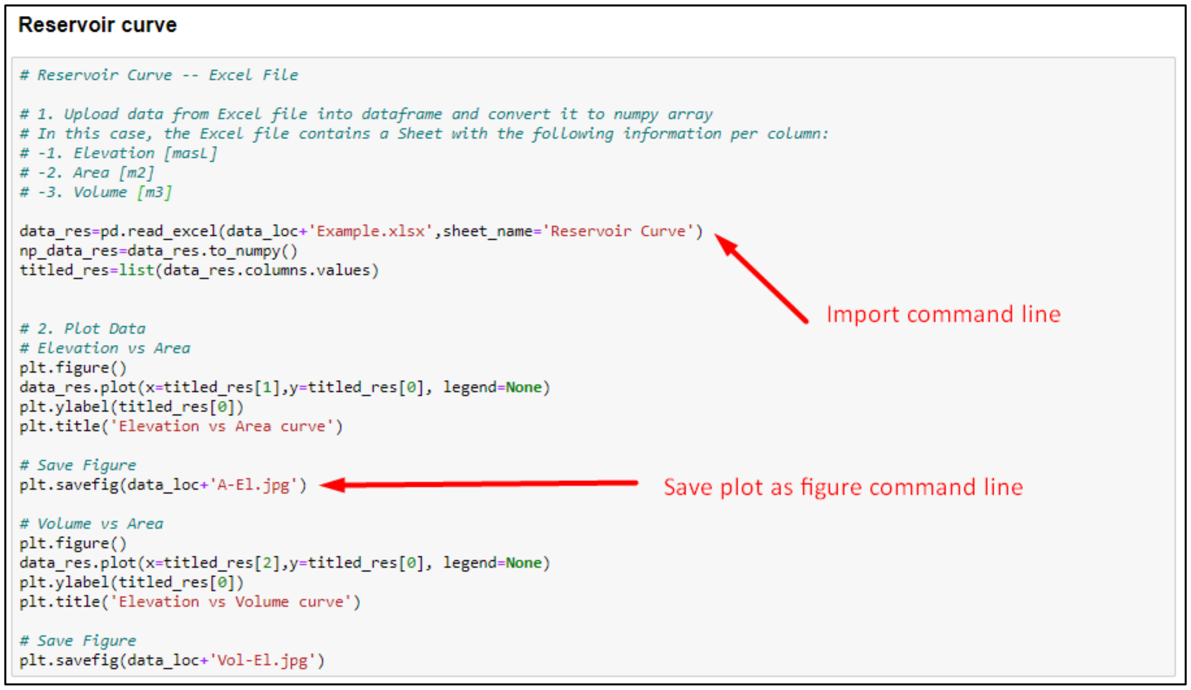
Figure 9-2. Set working directory

3. Input reservoir curves: Surface Area vs Elevation and Volume vs Elevation as Excel Files. Since the routing is solved using Equation (5), the surface area reservoir curve is the only one used along the Script. Table 9-1 displays an example of the imputed data (in this case the units are SI).

Elevation [masL]	Area [m^2]	Volume [m^3]
0.00	4046.86	0
0.15	4046.86	616.741
0.30	4046.86	1233.482
0.46	4046.86	1850.223
0.61	4046.86	2466.964

Table 9-1. Paired Data of Surface Area vs Elevation

The script uploads the files as dataframes and then it converts them into numpy arrays. The name of the Excel file as well as the sheet name must be inputted in the import command line. Plots of both reservoir curves are exhibited and can be stored as figures (Figure 9-3).



Reservoir curve

```
# Reservoir Curve -- Excel File

# 1. Upload data from Excel file into dataframe and convert it to numpy array
# In this case, the Excel file contains a Sheet with the following information per column:
# -1. Elevation [masL]
# -2. Area [m2]
# -3. Volume [m3]

data_res=pd.read_excel(data_loc+'Example.xlsx',sheet_name='Reservoir Curve')
np_data_res=data_res.to_numpy()
titled_res=list(data_res.columns.values)


# 2. Plot Data
# Elevation vs Area
plt.figure()
data_res.plot(x=titled_res[1],y=titled_res[0], legend=None)
plt.ylabel(titled_res[0])
plt.title('Elevation vs Area curve')

# Save Figure
plt.savefig(data_loc+'A-El.jpg')

# Volume vs Area
plt.figure()
data_res.plot(x=titled_res[2],y=titled_res[0], legend=None)
plt.ylabel(titled_res[0])
plt.title('Elevation vs Volume curve')

# Save Figure
plt.savefig(data_loc+'Vol-El.jpg')
```

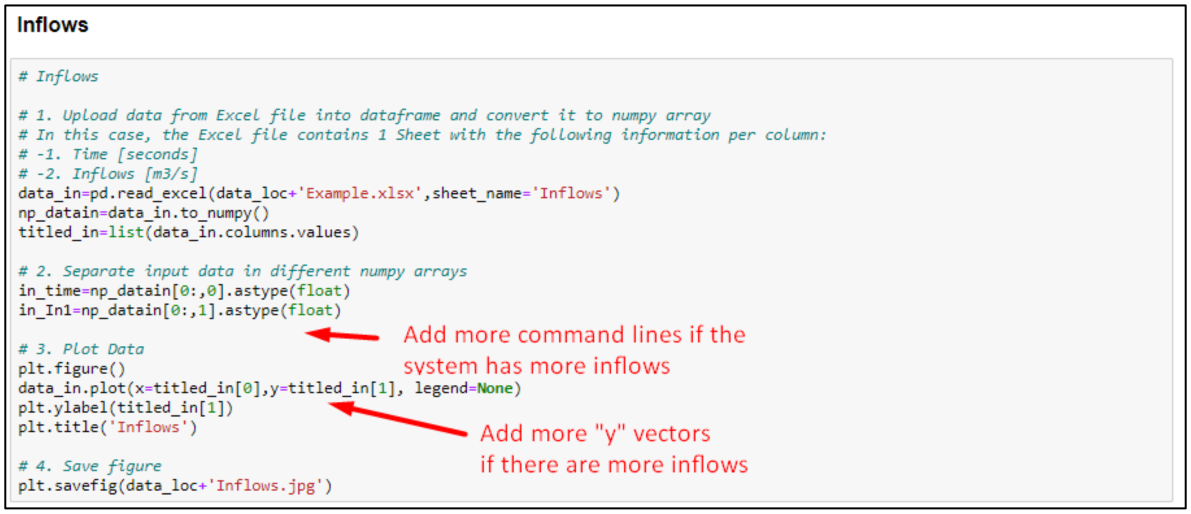
Figure 9-3. Script extract used to import, plot and save reservoir curve from Excel file

4. Input inflows as time series: Table 9-2 shows an example of inflow input data. The first column must always be the time in seconds while the second column contains the water inflows. More columns can be added if there are more inflows into the system. The number of columns vary depending on the analysed system. This can be changed by the user but implies changes in the routing programs, also performed by the user. If there is no need to have the inflows separated, it is recommended to add them all into one column.

Time [s]	Inflow [m^3/s]
0	0.000
600	1.699
1200	3.398
1800	5.097
2400	6.796

Table 9-2. Time series of inflows in the reservoir

The script uploads the files as dataframes and then it converts them into numpy arrays. The name of the Excel file as well as the sheet name must be inputted as explained in 3. Then, every column is separated into an individual array. Plots of inflow data series are exhibited and can be stored as figures (see Figure 9-4). Depending on the number of inflows, the user must add or delete command lines.



Inflows

```
# Inflows

# 1. Upload data from Excel file into dataframe and convert it to numpy array
# In this case, the Excel file contains 1 Sheet with the following information per column:
# -1. Time [seconds]
# -2. Inflows [m3/s]
data_in=pd.read_excel(data_loc+'Example.xlsx',sheet_name='Inflows')
np_datain=data_in.to_numpy()
titled_in=list(data_in.columns.values)

# 2. Separate input data in different numpy arrays
in_time=np_datain[0:,0].astype(float)
in_In1=np_datain[0:,1].astype(float)

# 3. Plot Data
plt.figure()
data_in.plot(x=titled_in[0],y=titled_in[1], legend=None)
plt.ylabel(titled_in[1])
plt.title('Inflows')

# 4. Save figure
plt.savefig(data_loc+'Inflows.jpg')
```

Add more command lines if the system has more inflows
Add more "y" vectors if there are more inflows

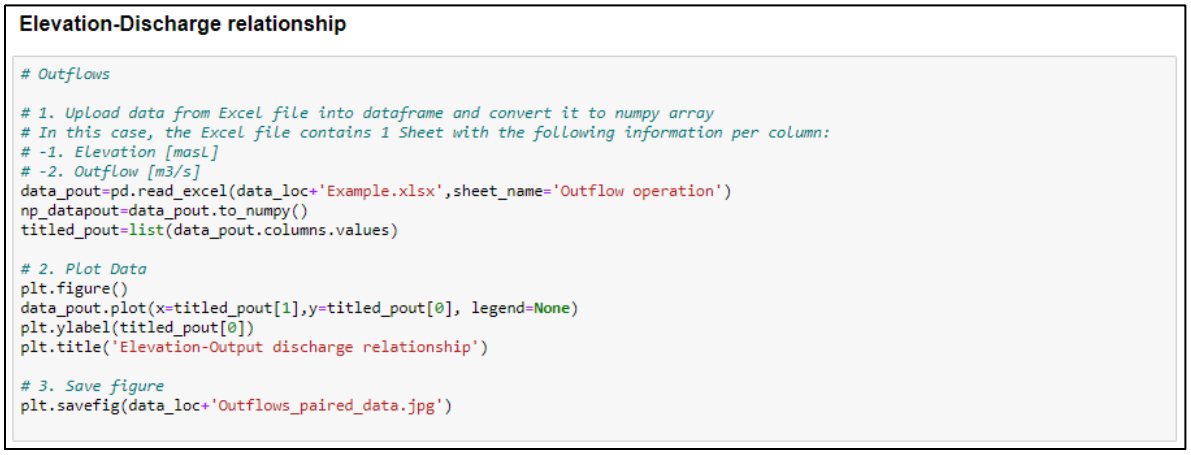
Figure 9-4. Script extract used to import, plot and save inflow data from Excel file

5. Input outflows as elevation-discharge relationship, i.e., paired data of outflow discharge as function of elevation for free spillways or bottom outlets (see Table 9-3).

H [m]	Q out [m^3/s]
0.00	0.00
0.15	0.08
0.30	0.23
0.46	0.48
0.61	0.85

Table 9-3. Paired data of outflows in the reservoir

The script uploads the files as dataframes and then it converts them into numpy arrays. The name of the Excel file as well as the sheet name must be inputted as explained previously. In this case, the columns are not separated since interpolation must be carried out in the routing process using both columns. Plot of the paired data are exhibited and can be stored as shown in Figure 9-5.

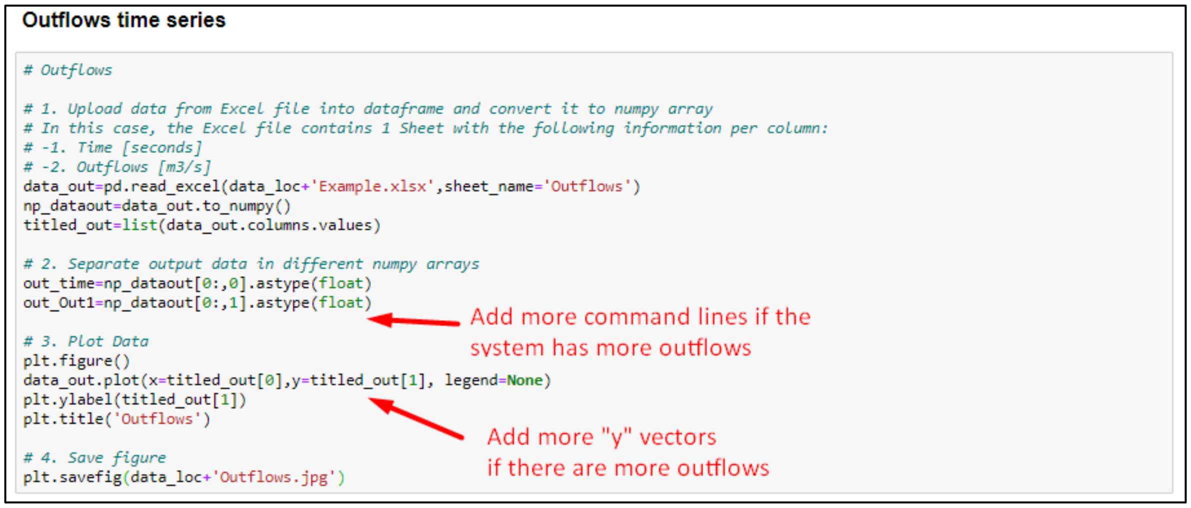


Elevation-Discharge relationship

```
# Outflows

# 1. Upload data from Excel file into dataframe and convert it to numpy array
# In this case, the Excel file contains 1 Sheet with the following information per column:
# -1. Elevation [masL]
# -2. Outflow [m3/s]
data_pout=pd.read_excel(data_loc+'Example.xlsx',sheet_name='Outflow operation')
np_datapout=data_pout.to_numpy()
titled_pout=list(data_pout.columns.values)

# 2. Plot Data
plt.figure()
data_pout.plot(x=titled_pout[1],y=titled_pout[0], legend=None)
plt.ylabel(titled_pout[0])
plt.title('Elevation-Output discharge relationship')

# 3. Save figure
plt.savefig(data_loc+'Outflows_paired_data.jpg')
```

Figure 9-5. Script extract used to import, plot and save outflow elevation-discharge relationship from Excel file

If the outflow data is provided as time series and not paired data of elevation-discharge, the cell with the script showed in Figure 9-5 must not be run. Instead, the code explained in 6 must be used. Input outflows as time series (if measured data is available – similar input format as shown in Table 9-2). As explained in 4, more columns can be added if there are more outflows out of the system. The number of columns vary depending on the analysed system. This can be changed by the user but implies changes in the routing programs, also performed by the user. If there is no need to have the outflows separated, it is recommended to add them all into one column.
The name of the Excel file as well as the sheet name must be inputted as explained in 3. Then, every column is separated into an individual array. Plots of outflow data series are exhibited and can be stored as figures (see Figure 9-6). Depending on the number of outflows, the user must add or delete command lines.


Outflows time series

```
# Outflows

# 1. Upload data from Excel file into dataframe and convert it to numpy array
# In this case, the Excel file contains 1 Sheet with the following information per column:
# -1. Time [seconds]
# -2. Outflows [m3/s]
data_out=pd.read_excel(data_loc+'Example.xlsx',sheet_name='Outflows')
np_dataout=data_out.to_numpy()
titled_out=list(data_out.columns.values)

# 2. Separate output data in different numpy arrays
out_time=np_dataout[0:,0].astype(float)
out_Out1=np_dataout[0:,1].astype(float)

# 3. Plot Data
plt.figure()
data_out.plot(x=titled_out[0],y=titled_out[1], legend=None)
plt.ylabel(titled_out[1])
plt.title('Outflows')

# 4. Save figure
plt.savefig(data_loc+'Outflows.jpg')
```

Add more command lines if the system has more outflows
Add more "y" vectors if there are more outflows

Figure 9-6. Script extract used to import, plot and save outflow data as time series from Excel file

6. Initial values: the user must input the initial water level, the time step in seconds and the number of time steps for the whole simulation.
7. Outflow data type: Elevation-discharge relationship
   In this section, the numerical methods were coded accounting for paired data outflow information. Afterwards, the routing is performed using the 4 methods and the results are stored in Excel and visualized graphically.
   7.1. Numerical methods – Euler and Modified Euler:
      Input Data: numpy arrays.
      - dt=time step for computations [s]
      - Tend=number of simulation steps [-]
      - H0=initial reservoir elevation [masL]
      - AEres=matrix of 2 columns - Elevation [masL] and Surface Area [m^2]
      - Qin1=vector - inflow [m^3/s]
      - Qout1=matrix of 2 columns - Elevation [masL] and Discharge [m^3/s]

      Output Data: numpy arrays.
      - y=vector - reservoir levels [masL]
      - t=vector - time [s]
      - A=vector - surface elevation [m^2]

- qin1=vector - inflow [m^3/s]
- qout1=vector - outflow [m^3/s]

Program:

1. The output vectors are created.
2. The values at time "0" for the output vectors are added (y[0], A[0] and qout1[0]).
3. The loop is created, starting computations for the second time step until the last time step.
4. The water levels and outflow discharges are computed for all time steps using Equations from (6) to (9).

7.2. Numerical methods – Runge Kutta 3rd order and 4th order:

Input Data: numpy arrays.

- dt=time step for computations [s]
- Tend=number of simulation steps [-]
- H0=initial reservoir elevation [masL]
- AEres=matrix of 2 columns - Elevation [masL] and Surface Area [m^2]
- Qin1= matrix of 2 columns - Time [s] and Discharge [m^3/s]
- Qout1=matrix of 2 columns - Elevation [masL] and Discharge [m^3/s]

Output Data: numpy arrays.

- y=vector - reservoir levels [masL]
- t=vector - time [s]
- A=vector - surface elevation [m^2]
- qin1=vector - inflow [m^3/s]
- qout1=vector - outflow [m^3/s]

Program:

1. The output vectors are created.
2. The values at time "0" for the output vectors are added (y[0], A[0] and qout1[0]).
3. The loop is created, starting computations for the second time step until the last time step.
4. The water levels and outflow discharges are computed for all time steps using Equations from (10) to (18).

7.3. Routing:

In this section, the 4 methods are runed, storing the results in several numpy arrays. To ease the results visualization, the numpy arrays are stored in Dataframes. One Dataframe is created for each numerical method and includes: time, reservoir level, inflow discharge and outflow discharge. Finally, a Summary Dataframe is created to compare the results of the 4 methods. The Dataframes are converted into an Excel file with 5 sheets, one for each Dataframe.

If more inflows entered the system, the codes must be modified to suffice the new requirements.

7.4. Results visualization:

A general plot is created to compare the results from the different numerical methods and is stored as Figure. Individual plots for each method can be created. In this case, the inflow and outflow hydrograph for the Runge Kutta 4th order method is coded. The user can use this script as an example to create its own plots.

8. Outflow data type: Time series
   In this section, the numerical methods were coded accounting for time series outflow information. Afterwards, the routing is performed using the 4 methods and the results are stored in Excel and visualized graphically. The code is quite similar to the one explained in 9. The only difference is that the input data for outflow changes from paired data to time series, influencing the code parts related to outflow. For the user, this only affects which parts of the script must be runed.
   8.1. Numerical methods – Euler and Modified Euler:
      - Outflow input data: vector [m^3/s]
   8.2. Numerical methods – Runge Kutta 3rd order and 4th order:
      - Outflow input data: matrix of 2 columns - Time [s] and Discharge [m^3/s]

Note: It's important to check consistency in the units of the input data. All discharges and areas must be on SI or English units. Do not mix the units.

9.1.1 Routing example using Python Script

To show that the Python script works properly, the example included by (Chow et al., 1988) was runed. The data in the example was given in English units; hence, conversion to SI units was carried out. Both types of outflow data were simulated, and the results were successful. This, the program was verified. Afterwards, the code was adapted to simulate the routing process through Punchiná reservoir, i.e., including more inflows and outflows.

- **Input Data**

Elevation [ft]	**Area [ft^2]**	**Elevation [m]**	**Area [m^2]**
0.0	43560	0.00	4046.86
11.5	43560	3.51	4046.86

Table 9-4. Reservoir curve
Source: Chow et al., 1988

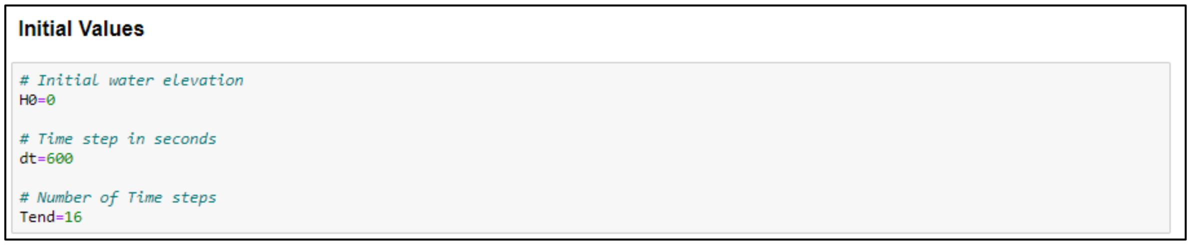
Time [min]	Inflow [cfs]	Inflow [m^3/s]	Outflow [cfs]	Outflow [m^3/s]
0	0.0	0.000	0.0	0.000
10	60.0	1.699	2.4	0.068
20	120.0	3.398	17.9	0.507
30	180.0	5.097	62.8	1.778
40	240.0	6.796	124.5	3.525
50	300.0	8.495	182.6	5.171
60	360.0	10.194	230.4	6.524
70	320.0	9.061	259.0	7.334
80	280.0	7.929	269.5	7.631
90	240.0	6.796	266.8	7.555
100	200.0	5.663	254.3	7.201
110	160.0	4.531	234.7	6.646
120	120.0	3.398	206.4	5.845
130	80.0	2.265	167.8	4.752
140	40.0	1.133	123.5	3.497
150	0.0	0.000	80.0	2.265

Table 9-5. Time series of inflows and outflows
Source: Chow et al., 1988

H [ft]	Q out [cfs]	H [m]	Q out[m^3/s]
0.0	0	0.00	0.000
0.5	3	0.15	0.085
1.0	8	0.30	0.227
1.5	17	0.46	0.481
2.0	30	0.61	0.850
2.5	43	0.76	1.218
3.0	60	0.91	1.699
3.5	78	1.07	2.209
4.0	97	1.22	2.747
4.5	117	1.37	3.313
5.0	137	1.52	3.879
5.5	156	1.68	4.417
6.0	173	1.83	4.899
6.5	190	1.98	5.380
7.0	205	2.13	5.805
7.5	218	2.29	6.173
8.0	231	2.44	6.541
8.5	242	2.59	6.853
9.0	253	2.74	7.164
9.5	264	2.90	7.476
10.0	275	3.05	7.787

Table 9-6. Elevation-Outflow Discharge relationship
Source: Chow et al., 1988

- **Initial values**



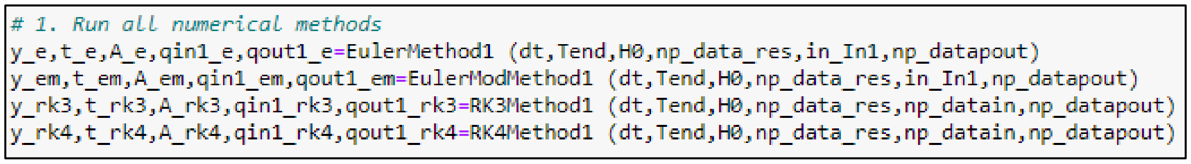
Initial Values

```
# Initial water elevation
H0=0

# Time step in seconds
dt=600

# Number of Time steps
Tend=16
```

Figure 9-7. Script extract where user inputs initial values

- **Running the program**

Each program (numerical method) receives the input data as numpy arrays and returns the routing simulation as numpy arrays. The code is shown in Figure 9-8.



```
# 1. Run all numerical methods
y_e,t_e,A_e,qin1_e,qout1_e=EulerMethod1 (dt,Tend,H0,np_data_res,in_In1,np_datapout)
y_em,t_em,A_em,qin1_em,qout1_em=EulerModMethod1 (dt,Tend,H0,np_data_res,in_In1,np_datapout)
y_rk3,t_rk3,A_rk3,qin1_rk3,qout1_rk3=RK3Method1 (dt,Tend,H0,np_data_res,np_datain,np_datapout)
y_rk4,t_rk4,A_rk4,qin1_rk4,qout1_rk4=RK4Method1 (dt,Tend,H0,np_data_res,np_datain,np_datapout)
```

Figure 9-8. Script extract where user runs the routing simulation using the 4 numerical methods

10Appendix 4: Multi-Objective Optimization Code

Reservoir Optimization

Implementation of the problem

Packages

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.dates as mdates
import datetime
from matplotlib.dates import DateFormatter
%matplotlib inline
```

Import data from Excel

```
# Set folder
data_loc='D:\\users\\mairm\\Documents\\TU DELFT\\Additional Thesis\\Python\\Optimization 10-11 FINAL VERSION\\'

# Read resevoir data
data_resae15=pd.read_excel(data_loc+'Reservoir Data.xlsx',sheet_name='AE_2015')
Ae=data_resae15.to_numpy()

# Read inflow data
datain=pd.read_excel(data_loc+'Output MeanDailyData 10-11.xlsx',sheet_name='Output')
np_datain=datain.to_numpy()
genSC_d=np_datain[:,4]
spillSC_d=np_datain[:,6]
inflow_d=np_datain[:,7]
date_d=np_datain[:,0]

# Read prices data
data_prices=pd.read_excel(data_loc+'Output MeanDailyPrices 10-11.xlsx',sheet_name='Output')
Prices_d=data_prices.to_numpy()
Prices_d=Prices_d[:,1]

# Read Water Levels
data_wl=pd.read_excel(data_loc+'Output MeanDailyLevel 10-11.xlsx',sheet_name='Output')
wl_d=data_wl.to_numpy()
wl_d=wl_d[:,1]

# Read Rating curve La Pesca
data_rcPesca=pd.read_excel(data_loc+'Routing Curve Ext La Pesca.xlsx',sheet_name='Output')
rcPesca_d=data_rcPesca.to_numpy()

# Read discharges at La Pesca
data_QallPesca=pd.read_excel(data_loc+'Output MeanDailyQPesca 10-11.xlsx',sheet_name='Output')
QallPesca_d=data_QallPesca.to_numpy()
QPesca_d=QallPesca_d[:,1]
QPescaopt_d=QallPesca_d[:,4]
```

Problem constraints

```
# Conversion Factor Function
def FC (Elev):
    if Elev<750:
        Elev=750
    elif Elev>780:
        Elev=780
    else:
        Elev=Elev
    fc=(-0.0001984389*Elev**2+0.3155122110*Elev-119.9051413867)
    return fc


# Define min and max conversion factors
FC_max=FC(781)
FC_min=FC(754)

# Define operational levels and max Spill during day
Hsw=775
Smax_bw=np.interp(Hsw,Ae[0:,0],Ae[0:,2])
Hsmax=781
Smax_aw=np.interp(Hsmax,Ae[0:,0],Ae[0:,2])
DSmax=Smax_aw-Smax_bw
Max_Spill_Day=DSmax/86400
Hsminop=754

# Define flood level
y_flood=126.3
y_war=126
```

```
# Define start and end of simulation (represents days in a year)
start=0
end=365
number=end-start

## Define input data from measurements and previous routing
inflow_d=inflow_d.astype(float)
inflow=inflow_d[start:end]
Prices_d=Prices_d.astype(float)
Prices=Prices_d[start:end]
wl_d=wl_d.astype(float)
wl=wl_d[start:end]
genSC_d=genSC_d.astype(float)
genSC=genSC_d[start:end]
spillSC_d=spillSC_d.astype(float)
spillSC=spillSC_d[start:end]
date=date_d[start:end]
QPesca_d=QPesca_d.astype(float)
QPesca=QPesca_d[start:end]
QPescaopt_d=QPescaopt_d.astype(float)
QPescaopt=QPescaopt_d[start:end]

# Simulation time step and end time of simulation
dt=86400
Tend=number

# Initial water level in reservoir
H0=wl[0]
```

```
## Setting upper and lower bound for optimization variables: generation flow and outlet flow
xu_array=np.zeros(number+number)
xl_array=np.zeros(number+number)

# For Generation
for i in range(0,number):
    xu_array[i]=1240/FC_max
    xl_array[i]=155/FC_min

# For Outlet
for i in range(number,number+number):  ##Check these values
    xl_array[i]=20
    xu_array[i]=30
```

Definition of problem

```
# Create class problem:
# Define variables used during optimization problem
# Define number of optimization variables, objective functions, inequality constraints and lower and upper bounds for
# the optimization variables

from pymoo.model.problem import Problem

class MyProblem(Problem):


    def __init__(self,Prices=Prices,inflow=inflow,Tend=Tend,dt=dt,Ae=Ae,H0=H0,Hsw=Hsw,Hsmax=Hsmax,Hsminop=Hsminop,
             Max_Spill_Day=Max_Spill_Day,Smax_bw=Smax_bw,y_war=y_war,rcPesca_d=rcPesca_d,QPescaopt=QPescaopt):
        super().__init__(n_var=2*number,
                         n_obj=3,
                         n_constr=5*number,
                         xl=np.array(xl_array),
                         xu=np.array(xu_array))
        #store custom variables needed for evaluation
        self.Prices=Prices
        self.inflow=inflow
        self.Tend=Tend
        self.dt=dt
        self.Ae=Ae
        self.H0=H0
        self.Hsw=Hsw
        self.Hsmax=Hsmax
        self.Max_Spill_Day=Max_Spill_Day
        self.Smax_bw=Smax_bw
        self.y_war=y_war
        self.rcPesca_d=rcPesca_d
        self.QPescaopt=QPescaopt
```

```
# Create def evaluate:
# Inside evaluate the optimization is performed, therefore, uses as inputs the stored variables
# define in class MyProblem (self)
# It also needs the population defined by X
# The routing, objective functions and inequality constraints are define inside evaluate
# The output of evaluate is a matrix where the columns are the simulated days and optimization variables
# and the rows represent the individuals found during the multi objective optimization

    def _evaluate(self, X, out, *args, **kwargs):

        # Define the input matrixes in the form [rows=solutions, cols=days]
        cols=len(inflow)
        rows=len(X)
        qin=np.zeros((rows,cols))
        qpesca=np.zeros((rows,cols))
        for j in range(0,cols):
            for i in range(0,rows):
                qin[i,j]=inflow[j]
                qpesca[i,j]=QPescaopt[j]

        level=np.zeros((rows,cols))
        level_exc=np.zeros((rows,cols))

        # Define the first wl value along with its surface area, volume, conversion factor and spill discharge
        y=np.zeros((rows,cols))
        y[:,0]=H0
        A=np.zeros((rows,cols))
        A[:,0]=np.interp(H0,Ae[0:,0],Ae[0:,1])
        FC_m=np.zeros((rows,cols))
        FC_m[:,0]=FC(H0)
        S=np.zeros((rows,cols))
        S[:,0]=np.interp(H0,Ae[0:,0],Ae[0:,2])
        Qspill=np.zeros((rows,cols))
        H_w=H0-Hsw
        if H_w<0:
            Qspill[:,0]=0
        else:
            Qspill[:,0]=(S[:,0]-Smax_bw)/dt
        # Routing
        # From routing, water levels and spill flows are estimated for every time step and every solution
        # X[i,j]: generated flow
        # X[i,j+cols]: outlet flow
        for i in range(0,rows):
            for j in range(0,cols-1):
                y[i,j+1]=y[i,j]+dt*(qin[i,j]-X[i,j]-X[i,j+cols]-Qspill[i,j])/A[i,j]
                A[i,j+1]=np.interp(y[i,j+1],Ae[0:,0],Ae[0:,1])
                S[i,j+1]=np.interp(y[i,j+1],Ae[0:,0],Ae[0:,2])
                FC_m[i,j+1]=FC(y[i,j+1])
                H_w=y[i,j+1]-Hsw
                if H_w<0:
                    Qspill[i,j+1]=0
                else:
                    Qspill[i,j+1]=(S[i,j+1]-Smax_bw)/dt


        # Check level at La Pesca for flood mitigation objective
        for i in range(0,rows):
            for j in range(0,cols):
                level[i,j]=np.interp(qpesca[i,j]+X[i,j]+X[i,j+cols]+Qspill[i,j],rcPesca_d[0:,0],rcPesca_d[0:,1])
                if level[i,j]>=y_war:
                    level_exc[i,j]=level[i,j]-y_war
                else:
                    level_exc[i,j]=0
```

```
        # Objective functions:
        # First Objective: Hydropower
        f1=0
        for n in range(0,cols):
            f1_1=X[:,n]*FC_m[:,n]*Prices[n]*1000*24
            f1=f1+f1_1
        f1 = -f1

        # Second Objective: Ecological flow at Guatapé river
        f2=0
        for n in range(0,cols):
            f2_1=X[:,cols+n]+Qspill[:,n]
            f2=f2+f2_1
        f2 = -f2

        # Third Objective: Flood Risk at La Pesca Village
        f3=0
        for n in range(0,cols):
            f3_1=level_exc[:,n]
            f3=f3+f3_1
        f3 = f3
```

```
        # Inequality and equality constraints (Equality constraints are written also as inequality constraints):
        # g1 and g2 set bounds for the water levels inside the reservoir to be inside the operational reservoir levels
        # g3 limits the daily spill discharge
        # g4 penalize the exceedance flood level instead of limiting the water level
        # g5 ensures that the last water level be the same as the first one (1 Jan=31 Dec)
        g1=np.zeros((rows,cols))
        g2=np.zeros((rows,cols))
        g3=np.zeros((rows,cols))
        g4=np.zeros((rows,cols))
        g5=np.zeros((rows,cols))

        for n in range(0,cols):
            g1[:,n]=-y[:,n]+Hsminop
            g2[:,n]=y[:,n]-Hsmax
            g3[:,n]=Qspill[:,n]-Max_Spill_Day
            g4[:,n]=level[:,n]-level_exc[:,n]-y_war

        # Epsilon helps to smoothed the constraint and helps to find feasible solutions
        epsilon=0.001
        for i in range(0,rows):
            g5[i,cols-1]=(y[i,cols-1]-H0)**2-epsilon

        # Stack all objective functions in one variable call F
        out["F"] = np.column_stack([f1, f2, f3])

        # Stack all inequality constraints in one variable call G
        out["G"] = np.column_stack([g1, g2, g3, g4, g5])

        # F, G and X (Objective functions, constraints and optimization variables) are the outputs from the optimization

vectorized_problem = MyProblem()
```

Initialization of the algorithm

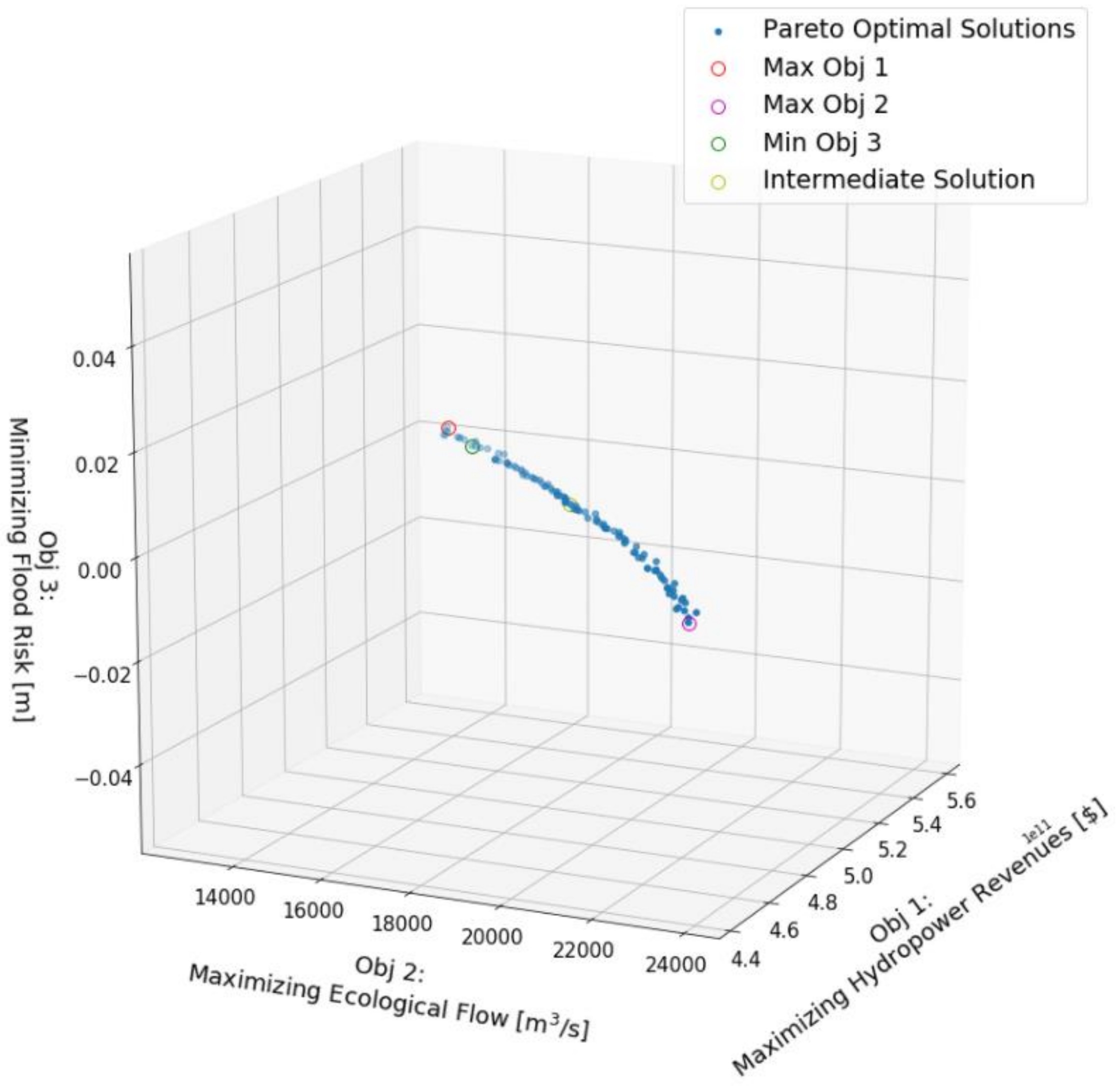
```python
# The multi-optimization uses the algorithm: NSGA-II -- Non dominated sorting genetic algorithm II
from pymoo.algorithms.nsga2 import NSGA2
from pymoo.factory import get_sampling, get_crossover, get_mutation
from pymoo.optimize import minimize

# Set algorithm characteristics: type of algorithm, population size, number of offsprings,crossover and mutation parameters

pop=600
offs=60
eta_cross=40
eta_mut=20


algorithm = NSGA2(
    pop_size=pop,
    n_offsprings=offs,
    sampling=get_sampling("real_random"),
    crossover = get_crossover("real_sbx", prob=1.0, eta=eta_cross, prob_per_variable=1.0),
    mutation=get_mutation("real_pm", eta=eta_mut),
    eliminate_duplicates=True  ## To be sure that mating produces unique offsprings
)
```

```python
# Define termination criteria

from pymoo.factory import get_termination

termination = get_termination("n_gen", pop)

# Funcitonal interface: minimize method.
# Calls the problem, the algorithm, the termination criteria
# It returns a Result Object that has the outcomes for Objective functions, constraints and optimization variables

res = minimize(MyProblem(),
               algorithm,
               termination,
               seed=None,
               save_history=True,
               verbose=False)
```

Results

```python
## Results can be printed by uncommenting the following line

# print("Best solution found: \nX = %s\nF = %s" % (res.X, res.F))

## Pareto optimal solutions

import matplotlib.ticker as mtick
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm

## The objective functions are separated in 3 arrays: Obj1, Obj2, Obj3

F=res.F
Obf1=(-F[:,0])
Obf2=(-F[:,1])
Obf3=(F[:,2])

# Get max index for the 3 objectives and an intermediate solution

max_index_Obf1=np.argmax(Obf1)
max_index_Obf2=np.argmax(Obf2)
min_index_Obf3=np.argmin(Obf3)
Obf1_x=Obf1[max_index_Obf1]
Obf1_y=Obf2[max_index_Obf1]
Obf1_z=Obf3[max_index_Obf1]
Obf2_x=Obf1[max_index_Obf2]
Obf2_y=Obf2[max_index_Obf2]
Obf2_z=Obf3[max_index_Obf2]
Obf3_x=Obf1[min_index_Obf3]
Obf3_y=Obf2[min_index_Obf3]
Obf3_z=Obf3[min_index_Obf3]
val=85
Obf123_x=Obf1[val]
Obf123_y=Obf2[val]
Obf123_z=Obf3[val]
```

```
# Plot pareto optimal solutions

fig=plt.figure(figsize=(15, 15))
ax=plt.axes(projection='3d')
ax.view_init(15,25)
ax.plot3D
ax.scatter3D(Obf1,Obf2,Obf3,label='Pareto Optimal Solutions')
ax.scatter3D(Obf1_x,Obf1_y,Obf1_z,s=100,color='r',marker='o',label='Max Obj 1',facecolor=(0,0,0,0))
ax.scatter3D(Obf2_x,Obf2_y,Obf2_z,s=100,color='m',marker='o',label='Max Obj 2',facecolor=(0,0,0,0))
ax.scatter3D(Obf3_x,Obf3_y,Obf3_z,s=100,color='g',marker='o',label='Min Obj 3',facecolor=(0,0,0,0))
ax.scatter3D(Obf123_x,Obf123_y,Obf123_z,s=100,color='y',marker='o',label='Intermediate Solution',facecolor=(0,0,0,0))

ax.set_xlim(Obf1[np.argmax(Obf1)],Obf1[np.argmin(Obf1)])
ax.set_xlabel('Obj 1: \nMaximizing Hydropower Revenues [$]',fontsize=18)
ax.set_ylabel('Obj 2: \nMaximizing Ecological Flow [m$^3$/s]',fontsize=18)
ax.set_zlabel('Obj 3: \nMinimizing Flood Risk [m]',fontsize=18)
ax.legend(loc='upper right',fontsize=19)
ax.tick_params(axis='x', labelsize=15)
ax.tick_params(axis='y', labelsize=15)
ax.tick_params(axis='z', labelsize=15,pad=10)

ax.xaxis.labelpad=30
ax.yaxis.labelpad=30
ax.zaxis.labelpad=30
```



```python
# Separating results and doing the routing per solution

X=res.X
sols=len(X)
cols=len(inflow)

# Create matrixes to store all values
# Each row represents a solution
# Each column is a day of the average year

QgenP=np.zeros((sols,cols))
QoutletP=np.zeros((sols,cols))
yres=np.zeros((sols,cols))
Qspill=np.zeros((sols,cols))
levels_Pesca=np.zeros((sols,cols))
levels_Pesca_exc=np.zeros((sols,cols))
Q_Pesca=np.zeros((sols,cols))

for i in range(0,sols):
    sol=i
    Q_gen=X[sol,0:cols]
    Q_outlet=X[sol,cols:cols+cols]
```

```python
    y=np.zeros(cols)
    y[0]=H0
    A=np.zeros(cols)
    A[0]=np.interp(H0,Ae[0:,0],Ae[0:,1])
    qin=inflow
    qpesca=QPescaopt
    FC_m=np.zeros(cols)
    FC_m[0]=FC(H0)
    S=np.zeros(cols)
    S[0]=np.interp(H0,Ae[0:,0],Ae[0:,2])
    Q_spill=np.zeros(cols)
    level=np.zeros(cols)
    level_exc=np.zeros(cols)
    H_w=H0-Hsw
    if H_w<0:
        Q_spill[0]=0
    else:
        Q_spill[0]=(S[0]-Smax_bw)/dt

    for j in range(0,cols-1):
        y[j+1]=y[j]+dt*(qin[j]-Q_gen[j]-Q_spill[j]-Q_outlet[j])/A[j]
        A[j+1]=np.interp(y[j+1],Ae[0:,0],Ae[0:,1])
        FC_m[j+1]=FC(y[j+1])
        H_w=y[j+1]-Hsw
        S[j+1]=np.interp(y[j+1],Ae[0:,0],Ae[0:,2])
        if H_w<0:
            Q_spill[j+1]=0
        else:
            Q_spill[j+1]=(S[j+1]-Smax_bw)/dt
```

```python
    for j in range(0,cols-1):
        y[j+1]=y[j]+dt*(qin[j]-Q_gen[j]-Q_spill[j]-Q_outlet[j])/A[j]
        A[j+1]=np.interp(y[j+1],Ae[0:,0],Ae[0:,1])
        FC_m[j+1]=FC(y[j+1])
        H_w=y[j+1]-Hsw
        S[j+1]=np.interp(y[j+1],Ae[0:,0],Ae[0:,2])
        if H_w<0:
            Q_spill[j+1]=0
        else:
            Q_spill[j+1]=(S[j+1]-Smax_bw)/dt

    for j in range(0,cols):
        level[j]=np.interp(qpesca[j]+Q_gen[j]+Q_spill[j]+Q_outlet[j],rcPesca_d[0:,0],rcPesca_d[0:,1])
        if level[j]>=y_war:
            level_exc[j]=level[j]-y_war
        else:
            level_exc[j]=0
```

```python
# Monthly results
# Store each solution on its on Dataframe and then estimate the mean monthly values
d = {}
monthly = {}
annual = {}
for i in range(0,sols):
    d[f'DayResults Solution{i}'] = pd.DataFrame({'Date':date,'Qgen (Optimization) [$m^3$/s]':QgenP[i,:],
                                               'Qgen (Measured) [$m^3$/s]':genSC,
                                               'Qoutlet (Optimization) [$m^3$/s]':QoutletP[i,:],
                                               'Qspill (Optimization) [$m^3$/s]':Qspill[i,:],
                                               'Qspill (Routing) [$m^3$/s]':spillSC,'Level (Optimization) [masL]':yres[i,:],
                                               'Level (Measured) [masL]':wl,'Level at La Pesca [masL]':levels_Pesca[i,:],
                                               'Exceedance Levels at La Pesca [masL]':levels_Pesca_exc[i,:],
                                               'QPesca (Optimization) [$m^3$/s]':Q_Pesca[i,:],
                                               'QPesca (Measured) [$m^3$/s]':QPesca})
    d[f'DayResults Solution{i}']=d[f'DayResults Solution{i}'].set_index(['Date'])
    monthly[f'Solution{i}']=d[f'DayResults Solution{i}'].resample('M').mean()
```

```
# Save data to Excel
# data_out=pd.DataFrame.from_dict(d)
with pd.ExcelWriter('D:\\users\\mairm\\Documents\\TU DELFT\\Additional Thesis\\Python\\
Optimization 10-11 FINAL VERSION\\Output TriObjective 10-11 FINAL 2_FINAL.xlsx') as writer:
     for i in range(0,sols):
        d[f'DayResults Solution{i}'].to_excel(writer,sheet_name=f'DayResults Solution{i}')
```

11 Appendix 5: Inflow discharges at San Carlos Project

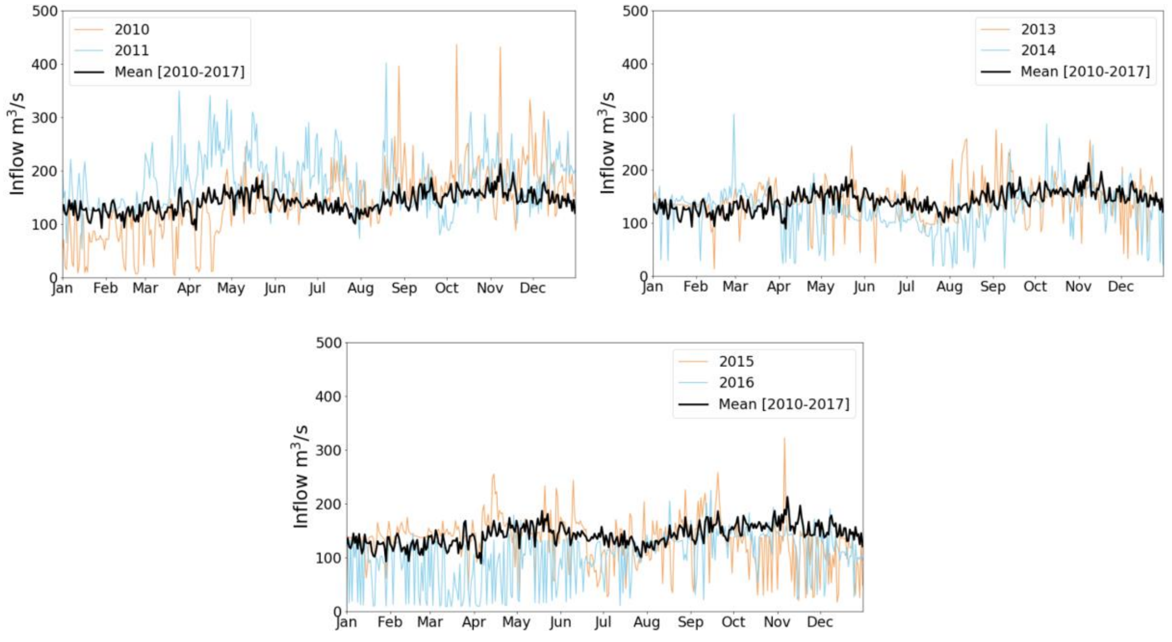


Figure 11-1. Inflow discharges at Punchiná reservoir. Top left: during La Niña Atmospheric Phenomenon, 2010-2011. Top right: during averaged weather conditions, 2013-2014. Bottom: during El Niño Atmospheric Phenomenon, 2015-2016

12 Appendix 6: Multi-objective optimization results. Period 2010-2011

12.1 Single Objective: Maximizing revenues from hydropower generation

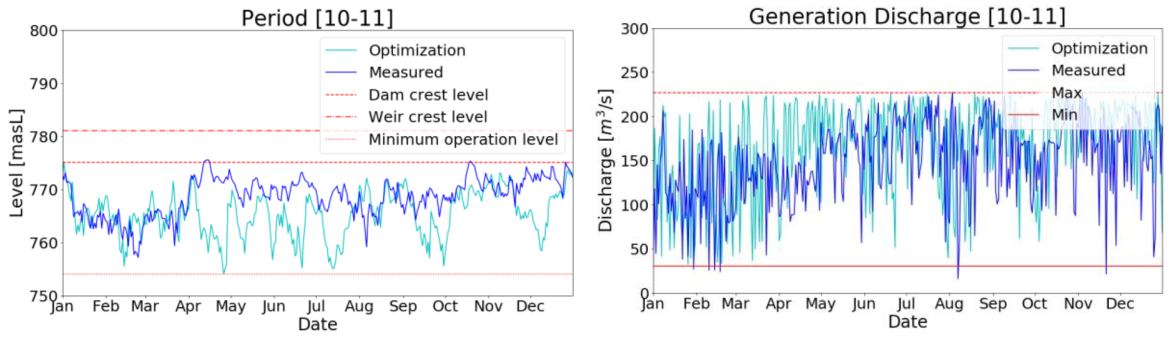


Figure 12-1. Daily single objective results during La Niña Atmospheric Phenomenon, 2010-2011

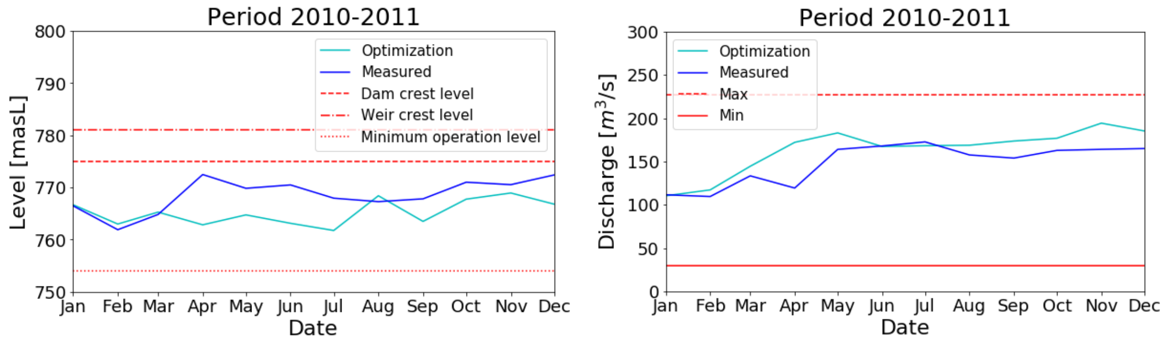


Figure 12-2. Monthly single objective results, period 2010-2011. Left: operation curve. Right: turbined discharges

12.2 Bi Objective: Maximizing revenues from hydropower generation and maximizing hydrological flow

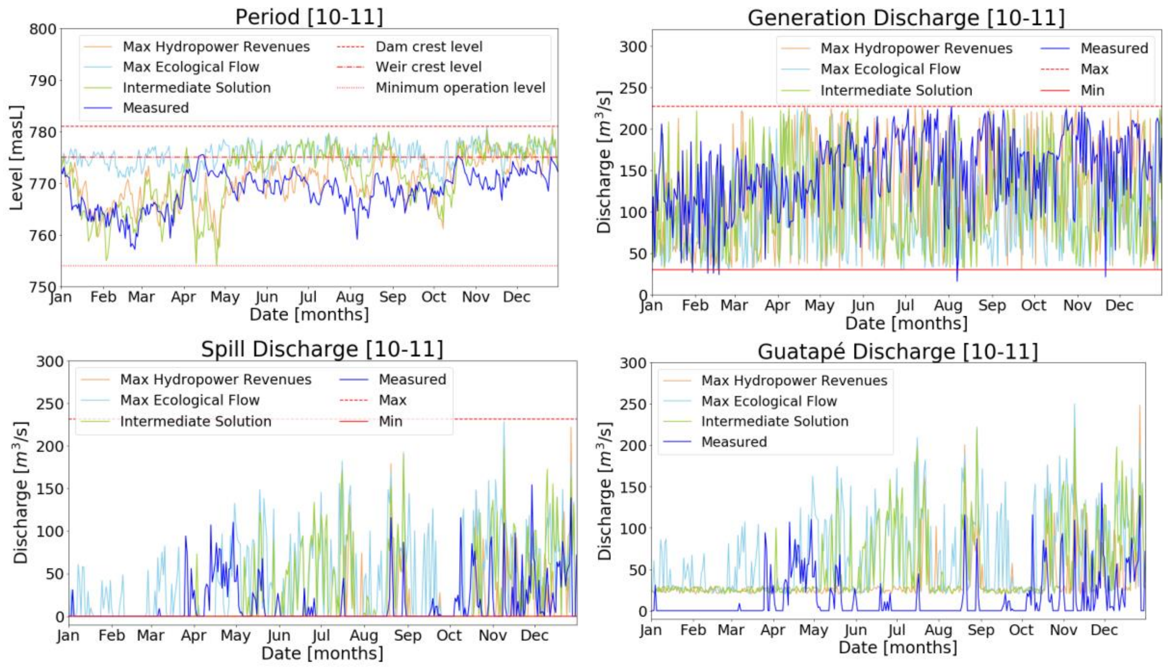


Figure 12-3. Bi objective optimization results during La Niña Atmospheric Phenomenon, 2010-2011, outlet flow in the range 20-30 m^3/s

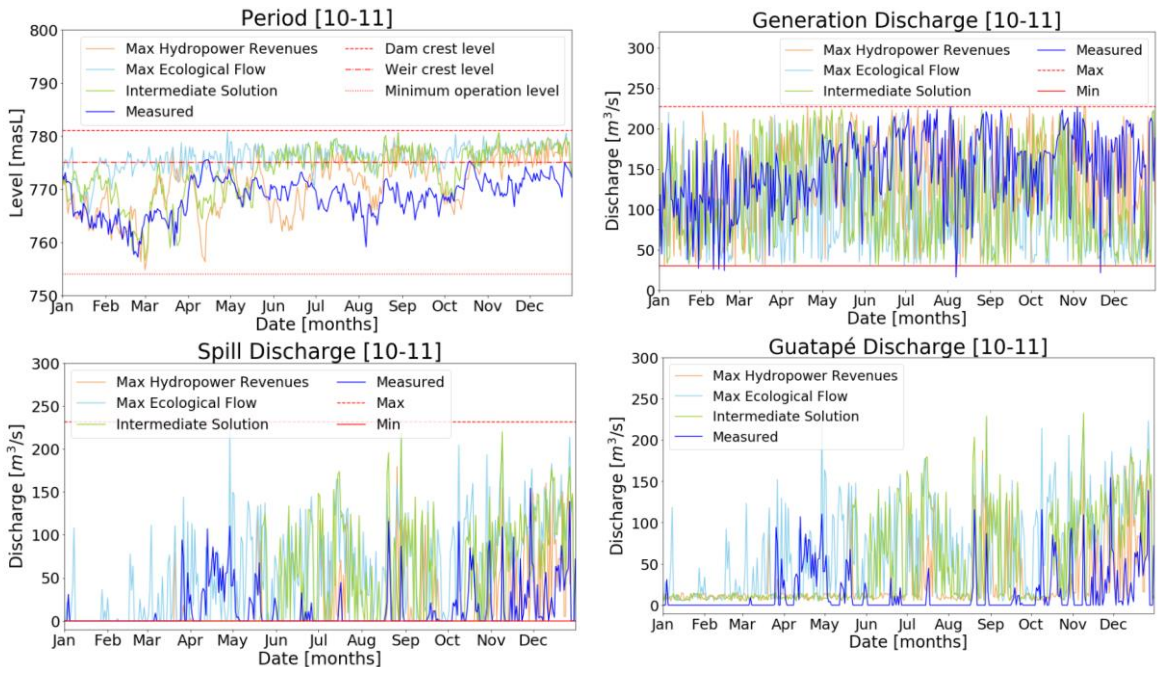


Figure 12-4. Bi objective optimization results during La Niña Atmospheric Phenomenon, 2010-2011, outlet flow in the range 5-15 m^3/s

13 Appendix 7: Multi-objective optimization results. Period 2013-2014

13.1 Single Objective: Maximizing revenues from hydropower generation

2

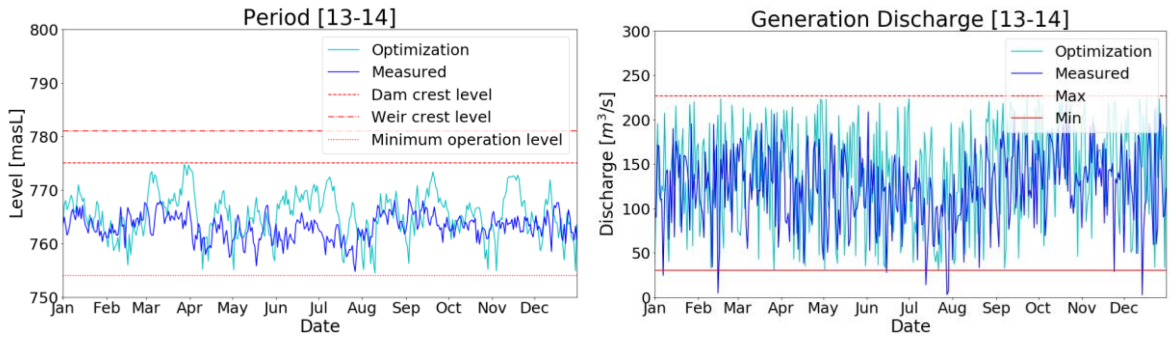


Figure 13-1. Daily single objective results during average conditions, 2013-2014

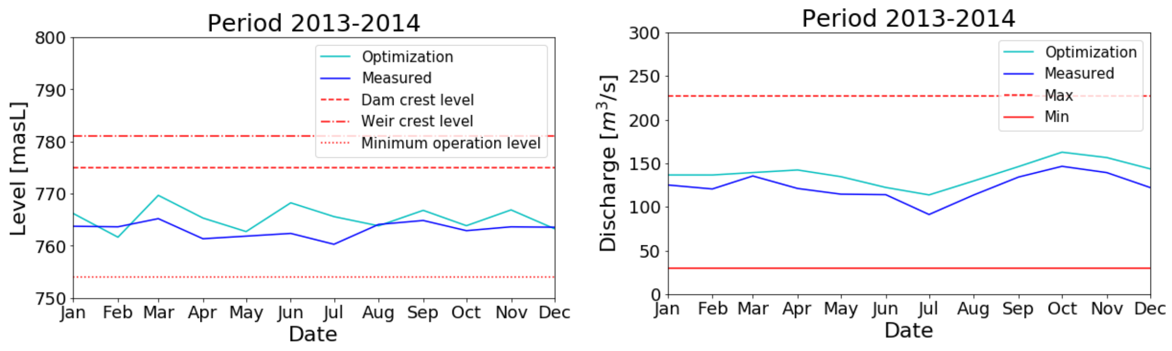


Figure 13-2. Monthly single objective results, period 2013-2014. Left: operation curve. Right: turbined discharges

13.2 Bi Objective: Maximizing revenues from hydropower generation and maximizing hydrological flow

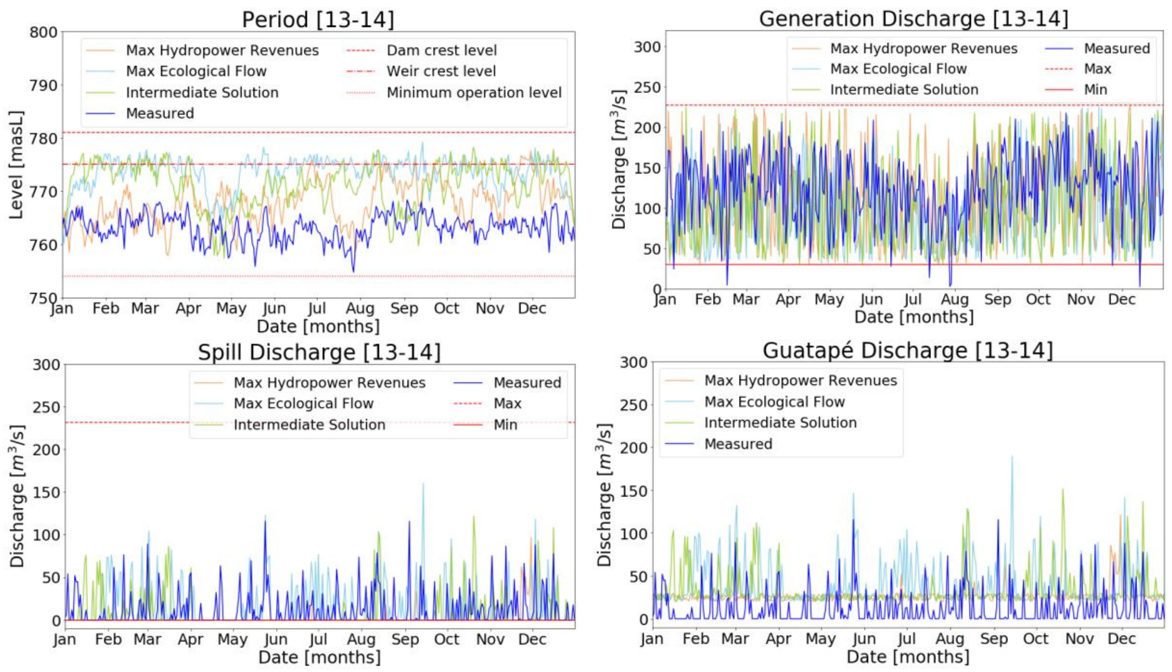


Figure 13-3. Bi objective optimization results during average conditions, 2013-2014, outlet flow in the range 20-30 m^3/s

14 Appendix 8: Multi-objective optimization results. Period 2015-2016

14.1 Single Objective: Maximizing revenues from hydropower generation

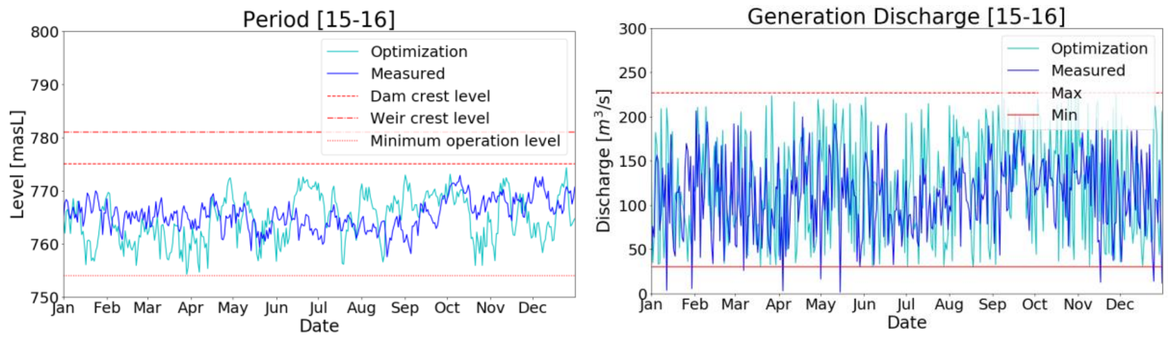


Figure 14-1. Daily single objective results during El Niño Atmospheric Phenomenon, 2015-2016

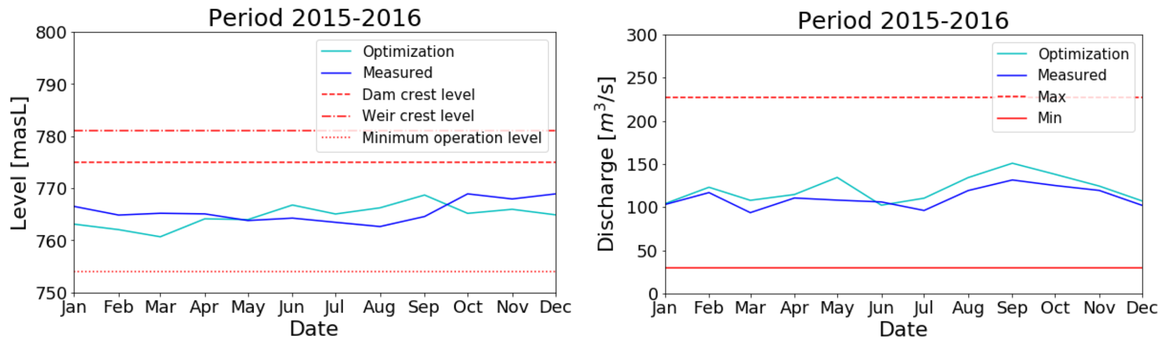


Figure 14-2. Monthly single objective results, period 2015-2016. Left: operation curve. Right: turbined discharges

14.2 Bi Objective: Maximizing revenues from hydropower generation and maximizing hydrological flow

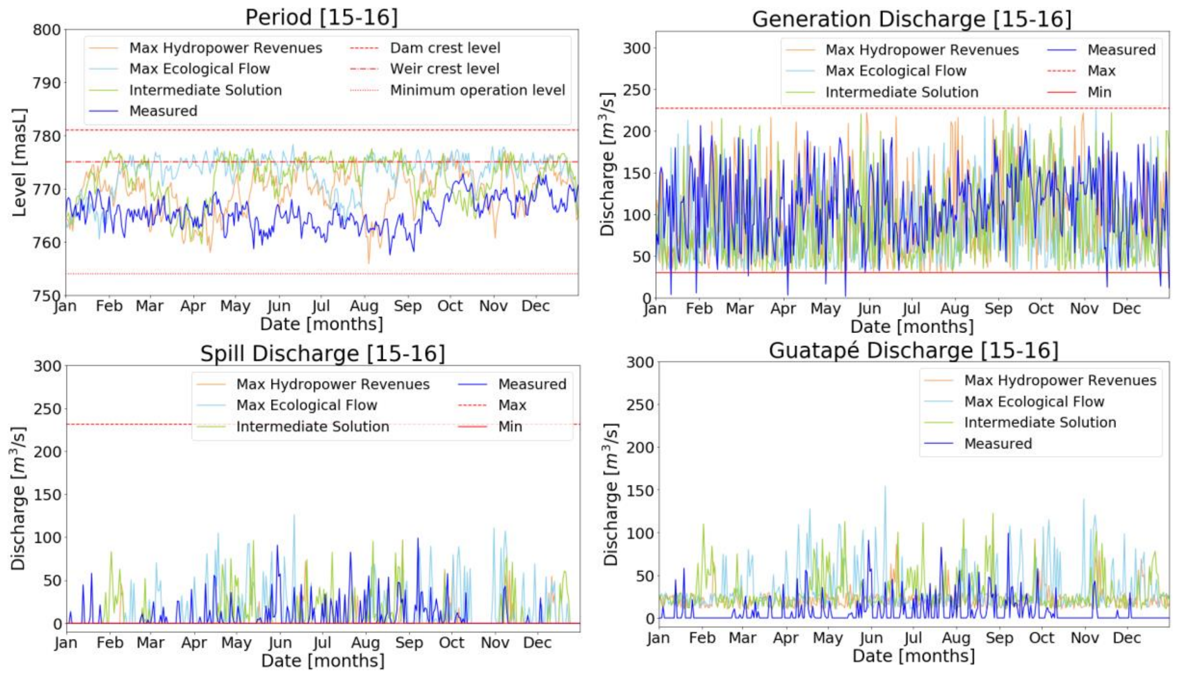


Figure 14-3. Bi objective optimization results during low flow conditions, 2015-2016, outlet flow in the range 10-30 m^3/s